The Dissertation Committee for Ganesh Krishnamoorthy Certifies that this is the approved version of the following dissertation:

A Framework for Utilizing Data from Multiple Sensors in Intelligent Mechanical Systems

Committee:

Delbert Tesar, Supervisor

Pradeepkumar Ashok

Ronald E. Barr

Richard H. Crawford

Dragan Djurdjanovic

Richard Hooper

A Framework for Utilizing Data from Multiple Sensors in Intelligent Mechanical Systems

by

Ganesh Krishnamoorthy, B.E.; M.S.E.

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

December 2010

Dedication

To my parents, for always being there for me.

Acknowledgements

I would like to express my sincere gratitude to my mentors Dr. Delbert Tesar and Dr. Pradeepkumar Ashok for their patient guidance of this research effort. I would also like to thank the members of my committee- Dr. Ronald Barr, Dr. Richard Crawford, Dr. Dragan Djurdjanovic, and Dr. Richard Hooper for their valuable suggestions. The Robotics Research Group management team of Dr. Mitchell Pryor, Ms. Janie Terrel, Ms. Betty Wilson and Ms. Malena Pomerleau-Petersen and the Mechanical Engineering department graduate coordinator, Mr. David Justh were instrumental in helping me negotiate the often complicated maze of protocols also known as doctoral studies.

I would be remiss if I didn't acknowledge the productive discussions I have had over the years with my RRG colleagues - Dr. Dinesh Rabindran, Dr. Jagadish Janardhan, Dr. Amit Kulkarni, Dr. Kyogun Chang, Dr. Stewart Vaculik, Dr. Oziel Rios, Kevin Kendrick and John Hall as well as the countless hours of work put in by Kevin Gentry to help develop the test set-up used in this work.

To all my dear friends- Dr. Krishnavikas Gudipati, Dr. Naveen Eluru, Dr. Abdul Rawoof Pinjari, Dr. Saurabh Bansal, Santhosh Padmanabhan, Dr. Vinod Viswanath, Gaurav Agarwal, Joe Prusaitis, Vaibhav Pattarkine, Sharanya Rao, Arvind Viswanath, Savitha Sridharan, Dr. Bharathkumar Thandri, Murali Narasimhan, Itisha Tyagi, Arun Radhakrishnan, Ashwini Gopinath, Arvind Rao, and Ashwini Talur- I will be eternally

grateful for your encouragement, for inspiring me to be a better person and for making Austin a home away from home for me.

Finally, to my parents, Dr. Krishnamoorthy and Dr. Uma Krishnamoorthy, I am forever indebted for their infinite reserves of patience. Your love, unwavering support and soothing words were something I could always count on even during the toughest of times throughout this endeavor.

This research was conducted under funding from the Office of Naval Research (Grant # N000140910427) and the Department of Energy (University Research Program in Robotics (URPR) Grant # DE-FG52-004NA25591). Their support is greatly appreciated.

A Framework for Utilizing Data from Multiple Sensors in Intelligent Mechanical Systems

Publication No._____

Ganesh Krishnamoorthy, Ph.D.

The University of Texas at Austin, 2010

Supervisor: Delbert Tesar

Electromechanical Actuators (EMAs) are being increasingly used in many applications. There is a need to augment good design of EMAs with continuous awareness of their operational capability and make them 'intelligent' for two key objectives: enhancing performance to address exigent task requirements and to track any changes from their 'as-built and certified' state for condition-based maintenance. These objectives are achieved using a decision making philosophy where the human system operator supervises EMA operation using performance criteria and decision surfaces; updated by in-situ measurement of the variables of interest via a suite of diverse sensors.

However, operational decisions made on the basis of faulty data could result in unwelcome consequences. With unexpected variations in a sensor's output from its anticipated values, the challenge is to determine if it indicates a problem in the sensor or the monitored system. In addressing this conundrum, it is also essential to account for the inherent uncertainties present in the values being analyzed. To this end, this dissertation presents the development of a novel Sensor and Process Fault Detection and Isolation

(SPFDI) algorithm. This provides a framework to utilize data from all the available sensors in a holistic manner to detect any faults in individual sensors or the system components concurrently. The algorithm uses a Bayesian network to model a system; populated with extensive empirical data. The probabilistic foundations of this method allow for incorporating and propagating uncertainties. The construction of a modular testbed and its Bayesian network are discussed in detail. Several design/ operational criteria have been proposed to aid in the creation of more usable networks in the future.

The SPFDI algorithm estimates multiple values for each measurand using different combinations of input variables and probabilistic inferencing. These values are compared against those indicated by the corresponding sensors; a difference between them is indicative of a potential problem. Quantitative indicators to track the condition of different system components and sensors, termed as belief values, are modified after each comparison. The final belief values obtained at the end of an iteration of the algorithm provide a definitive indication of the sources of anomalies in the observed data and can provide guidance to the operator on decisions such as whether or not to use data from a particular sensor for updating existing decision surfaces.

The representative examples and experimental results confirm the efficacy of the algorithm in detecting and isolating single as well as multiple sensor faults. The algorithm has also been found to be capable of distinguishing between sensor and system/process faults. Special categories of faults and factors that influence the execution characteristics and quality of results from the algorithm were also explored meticulously and suitable modifications have been suggested to enable the algorithm to continue to function effectively in these situations. To demonstrate the flexibility of the proposed SPFDI algorithm, its potential utilization in four broad classes of applications consisting of complex systems monitored by multiple sensors was also explored in this report.

Table of Contents

List of Tables

List of Figures

Chapter 1. Introduction

1.1. INTRODUCTION

Research efforts over the years at the University of Texas Robotics Research Group (UT-RRG) have been centered on creating the foundation for an open architecture of intelligent and reconfigurable systems- ranging from industrial manipulators for manufacturing to precision surgical robots; from mobile robotic systems to naval and aircraft subsystems, human rehabilitation, battlefield systems, etc. [Tesar, 2009]. At the core of this effort is the modular intelligent Electro-Mechanical Actuator (EMA)[1], which dictates the performance parameters of the overall system like stiffness, power, efficiency, etc. EMAs offer significant advantages over purely electrical (more power-dense), pneumatic or hydraulic systems (lower weight/volume, no complex piping/valves, reduced maintenance, etc.) with comparable or better performance characteristics (force/torque, speed, efficiency, noise). The objective is to develop an array of EMAs of different capabilities (high torque, ruggedness, quietness, fault tolerant, high precision, etc.) with standardized hardware and operating software, to populate the widest possible range of applications.

Figure 1-1: Electro-Mechanical Actuator

Sophisticated component-level design especially in the areas of prime movers [Ashok and Tesar, 2001], gear trains [Sigwald and Tesar, 2008], [Park and Tesar, 2005], and design configuration management [Vaculik and Tesar, 2008] has contributed immensely in increasing the performance of EMAs by orders of magnitude over many state-of-the-art options available commercially [Tesar, 2009]. However, even with such improvements, as applications in which EMAs are deployed demand ever-increasing performance, it is essential to use the resources available in an EMA to their utmost

[1] In this work, an EMA refers to a rotary actuator with an electric prime mover (switched reluctance or brushless DC motor), gear train, bearings, sensors, controller, etc. all contained in a compact, integrated package with standardized hardware and software interfaces for plug-and-play capability [Tesar, 2009].

potential. This can be achieved by making the EMA intelligent. The question arises- what exactly defines 'intelligence' in the context of EMAs.

1.2. THE 'INTELLIGENT EMA'

An intelligent EMA can be defined as one which has complete awareness of its operational capability at all times and has the ability to reconfigure its resources optimally to adapt to varying operational duty cycles or task requirements. Such generality of response can be imparted by the EMA controller and sensors along with novel control algorithms (in embedded software) [Tesar, 2009]. An intelligent EMA can be said to have the following characteristics [Tesar, 2009][Ashok and Tesar, 2007]:

- *Use of criteria-based Control:* Numerous performance criteria are used in conjunction with parametric models of the EMA, sensor-inferred models and end-user priorities to adapt to operational demands imposed by the application in which the EMA is deployed.
- *Reprogrammability and reconfigurability:* The functionality of EMA hardware/software can be altered easily based on extant task requirements[2].
- *Provision of fault tolerance when necessary:* When required, an intelligent EMA can utilize redundant resources[3] or use its available resources to continue operation under any partial or complete component-level faults. With these additional resources, the EMA can also provide more advanced operational capabilities like layered control (mixing physical scales of motion [Rabindran and Tesar, 2004]) or combined force-motion control (independent force/velocity priorities [Rabindran and Tesar, 2009])
- *Condition-Based Maintenance (CBM):* An intelligent EMA has the ability to continually monitor its performance, identify any incipient component degradation, and provide timely warnings about any impending malfunctions. This enables preemptive or condition-based maintenance to be carried out. CBM

[2] For example, altering motor phase angles for quieter operation [Ashok and Tesar, 2007]

[3] Redundant resources could exist as duplicate hardware within the EMA (e.g. dual gear trains) or in an analytical form, such as models/maps or heuristic knowledge.

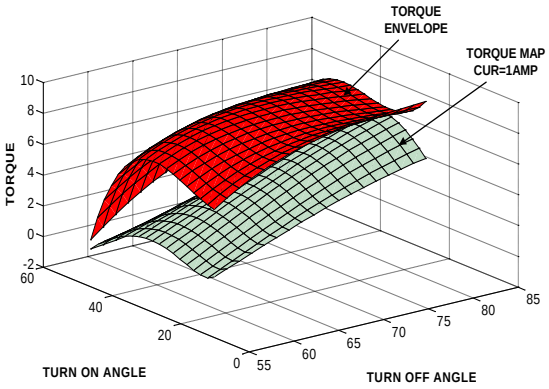
provides the benefits of reduced overall maintenance costs and a reduction in false alarms regarding the remaining useful life of an EMA [Hvass and Tesar, 2004] [Tesar, et. al, 2010].

The elemental requirement for EMA intelligence i.e. complete operational awareness can be addressed via two key sources of information regarding the EMA capabilities. The first is comprehensive knowledge/parametric models of the performance capability of a new (as-built) EMA over the complete range of its operating points, represented in the form of performance maps and envelopes (Section 1.2.1). The second is an updated profile of the same capabilities assessed in real-time during operation, by in-situ sensors (Section 1.2.2).

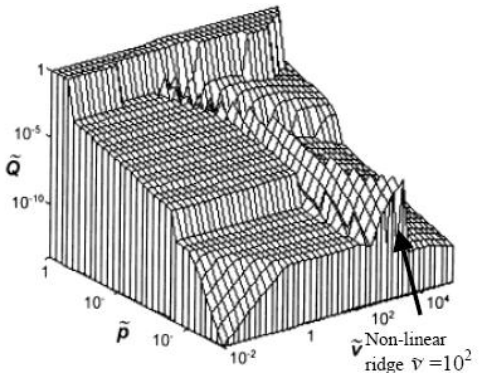
1.2.1. Performance Maps and Performance Criteria

[Janardhan and Tesar, 2008] emphasize that nameplate specifications of EMA performance are nominal metrics and valid only for a specific operating state (e.g. torque-speed ratings of an electric motor at no load). However, in reality, an EMA operates over a broad range of conditions and such single-point specifications do not provide much information regarding how the system is likely to behave under different circumstances. A solution to this issue is often found in the form of analytical models/expressions for different parameters of interest in the system that can be used over the entire operating domain. For instance, in a brushless DC motor, torque T is often estimated via a linear relation $T = K_t I$ (K_t = torque constant, I=current). However, such analytical descriptions often neglect the nonlinear nature and the high degree of coupling between the various EMA operational parameters[4]. For instance in the example above, the value of K_t is also influenced by factors like operating speed, magnetic saturation, temperature, etc. [Yoo and Tesar, 2004]. These simple models are therefore unable to provide accurate operational estimates. With aging and extended operation, these relations may undergo further subtle changes leading to incorrect estimates of values of interest. Hence, rather than using single-point measures to characterize system performance there is a need for

[4] Power parameters (e.g. torque/speed), operating parameters (e.g. bandwidth) and environmental parameters (e.g. temperature)



Example of a Performance Map and Performance Envelope for a Switched Reluctance Motor [Ashok and Tesar, 2007]



Example of a Gear Train Performance Map [Podra and Andersson, 1999]

Figure 1-2: Examples of Performance Maps

other means to characterize the behavior of different parameters over the complete operating range of an EMA. This led to the development of the concept of performance maps and envelopes [Tesar, et. al, 2005].

A performance map represents the functional relation among the control, reference and dependent parameters of an EMA for all its achievable operating points [Tesar, et. al., 2005][5]. The control parameters are the inputs that govern the EMA operation (e.g. voltage, phase turn-on/off angles for motor commutation, etc.). The reference parameters are those that cannot be directly controlled and usually represent the state variables (e.g. speed, acceleration, torque, etc.). The dependent parameters are those which do not affect the other control or reference parameters and can be considered as 'effect' variables (for example, noise generated by the EMA). A performance envelope may represent the extreme limits of performance i.e. the limits to which the EMA operation can be pushed for short periods on demand or for known periods at specified performance levels without breakdown or an envelope may be an algebraic combination of several maps. A set of such maps can be

[5] Researchers have used concepts similar to performance maps for diverse applications- [Omekanda, 2003] to determine optimal operating points for a switched reluctance motor (from torque efficiency and ripple perspective), [Paganelli et. al, 2001] for automobile powertrain performance optimization, [Sainsot and Flamand, 1994] to characterize the power losses in gear bearings under different loading conditions, etc.

defined for each component of the EMA (Figure 1-2) or the EMA as a complete system, using the functional form described in [Tesar, et. al., 2005] and [Ashok and Tesar, 2007]:

$$PM_{(*,m)}: f(p_1, p_2, p_3) = K \qquad \text{Eq. (1-1)}$$

where *PM* indicates performance map, * represents one of {*p, g*, *b*, *c*} (*p*-prime mover*, g*-gear train, *b*-bearings, and *c*-controller); m is an index for the number of maps, p_1, p_2 are control or reference parameters, p_3 is a dependent or a reference parameter or a performance criterion, and *K* is a constant. Such performance maps can be generated by:

- Empirical testing over the entire EMA operating range [Janardhan and Tesar, 2008]
- Combining related performance maps of individual EMA components (motor, geartrain, bearings, etc.) into an overall EMA performance map [Ashok and Tesar, 2007]
- Via analytical methods [Omekanda, 2003]

Irrespective of the method used, the sheer volume of information contained in a performance map presents a significant challenge in terms of conveying it to the end-user for use in operational decision-making[6]. The data points for the maps can be compactly stored as look-up tables or as fitted regression equations. These methods often become 'black-box' models and do not convey an insightful interpretation to the end-user as to how certain operational parameters vary with respect to the independent control variables [Janardhan and Tesar, 2008]. However, a visual representation (of relevant performance measures over the complete range of operating conditions) can provide a more intuitive interface that can not only be used by a non-expert system operator to make appropriate operational decisions based on existing operating conditions but also retain sufficient granularity of information for an expert user to analyze EMA performance over a broad range of the parameters of interest or to perform diagnostics [Ashok and Tesar, 2007]. For easy visualization, performance maps are represented as three dimensional surfaces

[6] Hence, [Ashok and Tesar]also refer to these maps as '*decision surfaces'*. However, performance maps need not be strictly restricted to the operational phase of a system. They may also be used to assist decision-making during the design process as described in [Vaculik and Tesar, 2008]

with p_1 and p_2 typically represented along the X/Y axes and p_3 on the Z axis (Figure 1-2) [Tesar, et. al, 2005][7].

In order to enable practical application of these maps in operational decision-making for any system, user-defined/ application specific criteria are needed (these are usually represented on the Z axis of a map). [Hooper and Tesar, 1994] outlined various properties that were desirable while defining similar criteria for system level decision-making in robotics such as physical significance of the criteria, continuous variation over the operational range of interest, computational efficiency, task independence, finiteness of magnitude, etc. amongst others. Based on these guidelines, the works of [Scott and Tesar, 1999], [Turner and Tesar, 2000], [Hvass and Tesar, 2004], [Yoo and Tesar, 2004] and [Janardhan and Tesar, 2008] led to the development of more than 40 different criteria to characterize EMA performance (e.g. rise-time, efficiency, current/voltage saturation, operational margin, torque ripple, velocity ripple, health-margin, backdrivability, output oscillation, remaining useful life, noise, vibration, temperature, etc., see [Ashok and Tesar, 2007]).

In addition to the development of numerous performance criteria, there was also a need to identify a few simple measures that could be derived from the available performance maps[8] and used to represent the criteria across the entire operational space of the EMA. The goal in this case was to provide a greater understanding of such criteria to the end user as well as to determine critical operating points on the map (especially when the duty cycle of an EMA requires it to function close to its operational margins). The use of such measures or norms (single-number extracts derived from the maps) was first demonstrated in [Hvass and Tesar, 2004] for Condition Based Maintenance (CBM) of EMAs. [Ashok and Tesar, 2007] further advanced the practice to develop a formalized decision making framework (using Bayesian networks) intended to address a wider set of

[7] A performance map can also be a hyper-surface i.e. a surface with dimensionality greater than 3, as described in [Hvass and Tesar, 2004]. With a three dimensional representation, any or all axes of a map can be reference parameters but a map can have only one dependent parameter (on the Z axis).

[8] Via relatively simple mathematical operations such as integration, differentiation, summing, differencing, averaging, normalization, etc. as described in [Ashok and Tesar, 2007] and [Hvass and Tesar, 2004]

objectives. Based on simple analytics, ten norms (monotonicity, maxima, minima, volume under the map, etc.) to characterize any performance map were developed. [Ashok and Tesar, 2007] also provide examples using these criteria/norms to combine different maps into performance envelopes as well as combining multiple operational criteria into a single map in order to provide optimum EMA performance over all those criteria. The authors assert that the proposed methodology could be used to address any operational requirement in concert with (or in some applications even supersede) traditional control methodologies.

The maps, criteria and norms, envelopes, etc. described thus far represent the first step towards achieving complete nonlinear operational awareness. But this is knowledge in a feedforward sense i.e. they represent how an EMA can be expected to perform under certain specific sets of control parameter conditions over the entire operating range. To complete the feedback cycle, this knowledge needs to be periodically updated with reliable and accurate information regarding all the parameters of interest in the application, reflecting the actual extant conditions of the EMA (for control purposes as well as to track the health of the various EMA sub-components using CBM algorithms).

1.2.2. Multisensor Architecture for EMAs

The ability of an EMA to respond adequately to a range of operating requirements or duty cycles is dictated chiefly by the embedded controller's capacity to accurately infer its present operating condition and capabilities, in order to decide on a suitable control strategy. Conventional EMA control schemes often rely on a small set of sensors for feedback (variables such as position/speed, current and voltage are most commonly used in many applications). In some specialized applications (such as robotics, aerospace applications, etc.) or in a testing/certification environment, additional sensors for variables such as temperature, torque, vibration, acoustic noise, etc. may also be used. Increased cost and additional design challenges in integration are often cited as the reasons why sensors in addition to those strictly needed for operation are not integrated into most commercial EMAs. The number of sensors is kept to a minimum, mainly to

reduce the overall costs. The perception is that adding more sensors to any system adds additional locations where single-point failures may occur, since even state-of-the-art sensors are susceptible, to some extent, to factors like temperature, shock/vibrations, electromagnetic interference, humidity, connector/cabling failures, etc[9]. However, adhering to such limited operational information especially in applications that demand a high level of EMA performance (for example, actuating control surfaces in fighter aircraft [Koran and Tesar, 2008], precision robotics, etc.), not only throttles its capabilities but can also prove catastrophic in case of sudden sensor failures or degradation since it may lead to eventual overall system failure.

With a limited set of internal sensors, the ensuing awareness of an EMA's actual operating conditions is also limited. This typically leads to a restriction of the EMA operation within a conservative range specified by the manufacturer (i.e. the rated specifications) with a view to avoid any deterioration of its sub-components (for example, operating the electric motor using a low current to prevent over-heating and avoid any damage to its windings). Almost always this means that, the EMA's capabilities are under-utilized[10], when they can actually be pushed beyond these nominal ranges for short periods of time when the application demands it (or in an emergency situation). There is thus a clear need to comprehensively track in real-time, the actual variation in the EMA variables vis-à-vis the 'as-built' or expected behavior of those parameters under the prevalent operating conditions. This is to account for any unaccounted changes in the available system models (i.e. the performance maps for each of those parameters).

Given the multitude of possible maps (as described in [Tesar, 2005] and [Ashok and Tesar, 2007]), this calls for the use of an extensive suite of diverse, in-situ sensors

[9] Other factors deterring the inclusion of multiple sensors are the lack of appropriate sensors and the available computation/ signal processing capabilities to handle multiple streams of data. Though the desired performance (range, resolution, etc.) can be found in off-the-shelf sensors, often the limiting factor is the lack of an appropriate form factor to allow for compact integration in an EMA. This in turn necessitates custom-designed solutions. This is often the case for output position sensors or torque sensors used in EMAs [Tesar, 2009]. However, with constant innovations and modern sensor and data acquisition/processing/computing technologies, these concerns have been alleviated to a great extent.

[10] This often leads to EMAs being over-designed for an application, leading to bulkier and costlier systems

that can be compactly integrated into an EMA[11]. As will be seen during the course of this work, the benefits of including multiple sensors far outweigh the risks; especially, since the enabling sensor technology exists. The issue of cost can also be justified in case of high-value applications where reliability (uninterrupted data availability), accuracy and extensiveness of available information can mean everything. To determine the set of measurands/sensors that are applicable across a range of applications, a detailed analysis of factors affecting the behavior of the various EMA components (which in turn has an effect on the overall EMA performance) was carried out and a ten-sensor architecture (position, velocity, acceleration, torque, temperature, noise, vibration, current, voltage, magnetic field) has been proposed to provide complete awareness and comprehensively track variations in the different EMA characteristics (refer Appendix A).

The advantages of integrating several diverse sensors in an EMA can be examined in context of the characteristics desired of an intelligent EMA (as outlined in the beginning of Section 1.2) as follows:

I. *Multi-Criteria Decision Making*: With additional sensors monitoring measurands (like temperature, noise, etc.) other than those strictly needed for EMA operation (such as position, current, etc.), it is possible to specify and utilize additional operational criteria (e.g. minimum acoustic noise) as benchmarks to quantify and improve the performance of an EMA. If redundant resources are available to expand the architecture of EMAs (such as multiple prime movers or gear trains [Tesar, 2009]), appropriate criteria may be used to judiciously allocate the available resources to meet a much broader set of task demands (for instance, as described in [Rabindran and Tesar, 2009] to independently allocate force/torque priorities) or address partial component failures (where the embedded controller can reconfigure alternate physical pathways in the EMA to accomplish the task). The concept is analogous to the approach used in robotic systems where multi-criteria decision making is often used to exploit any kinematic redundancy

[11] It is obvious that quality information from the same diverse group of sensors is also needed to generate the performance maps in the first place.

available in a robotic manipulator for fault tolerance, recovery, etc. [Cetin and Tesar, 1998] [Cocca and Tesar, 2000].

II. *Operational Fault Tolerance*: In any system, sensor feedback is used to determine the appropriate control action to dictate the system operation. The prime concern is therefore that any loss of data from one or more sensors should have the least disruptive effect on the overall system operation (or it should not lead to a complete failure of the system in the worst case). Operational fault tolerance in the context of sensors can be defined as the ability to detect, identify and accommodate partial or complete failures. This can allow a system to continue to function at an acceptable level of performance or at least have a 'limp-home' capability in safety-critical situations without requiring urgent intervention.

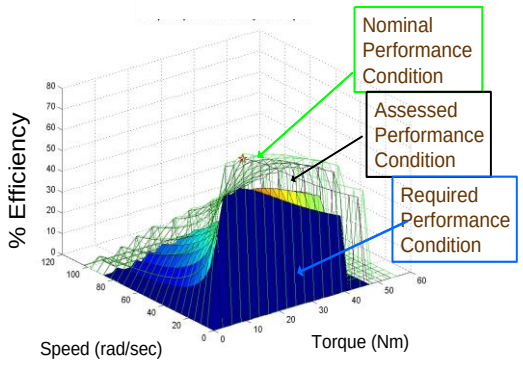
Although the easiest way to achieve this objective is via hardware redundancy for each sensor, such an approach adds greater complexity, cost and weight to EMAs that are already hard-pressed in terms of having to integrate many components in a dense package. Hence, it is often not feasible especially if the EMA is used in applications where volume/weight may be at a premium[12]. Using sensors for a diverse set of measurands and correlating the different variables through appropriate models, it is possible to detect the signs of degradation/failure in any sensor and provide alternative pathways to infer the different variables of interest via functional redundancy[13] (this aspect is explored further in Chapter 5). Thus, under nominal operating conditions, the performance of an EMA with additional sensors is either improved (due to more comprehensive evaluation of the EMA operational capability) or at least remains the same. But in off-nominal conditions, the availability of additional sensors can

[12] Despite its drawbacks, hardware redundancy is still the preferred approach especially in applications where human safety is at stake (such as in aircraft where quad-redundancy in critical systems is still prevalent [Tesar, 2009]). Depending on the configuration of the different sub-components within the EMA and the resulting geometrical constraints, it may still be possible to incorporate hardware redundancy for some sensors such as temperature, current, and voltage sensors, etc.

[13] This may be as simple as deriving the value of a variable A using the values of other variables B, C, D, etc. using available relations (for instance, $A=f(B)$, $A=g(C)$, $A=h(D)$, etc.) and using a voting scheme.

make a major difference to operation by enabling information fault tolerance.

III. *Actuator Condition Monitoring*: For EMAs used in applications like aerospace, tele-operated robotic systems used in nuclear facilities, etc. reliability of the system and ensuring safety of the users involved are two of the prime objectives. There is an evident need in these applications for techniques that can provide advanced warning of any potential degradation in performance and prevent any minor faults that may originate in EMA components like bearings, motor, etc. (due to quality of manufacturing and materials used, wear and tear of contacting surfaces, environmental effects like heat, humidity, excessive vibrations, etc.) from developing into major faults. However, signaling a 100% EMA failure when signs are noticed that some component(s) are not performing at desired levels, neglects any residual capability an EMA may have to perform the task at hand. A scheduled/periodic maintenance approach may result in EMAs being replaced earlier than necessary, leading to higher lifecycle costs. Adopting a continuous monitoring strategy for EMAs is a more prudent alternative, wherein its performance capability is continuously evaluated and it remains in service until it is estimated that it is no longer capable of meeting basic/essential task demands.



[Hvass and Tesar, 2004]

Figure 1-3: Condition-Based Maintenance

In the Condition Based Maintenance (CBM) methodology proposed by [Hvass and Tesar, 2004], vital EMA variables are continuously monitored for signs of degradation. The occurrence of faults in any of the EMA components leads to a change in the model parameters represented by its corresponding map

(Figure 1-3 represents the changes occurring in the efficiency map of a brushless DC motor due to winding faults). The residuals between the sensor-inferred values and the estimated values (from the map) are used to determine the extent of degradation and to calculate the remaining useful life, the % health margin (the relative ability of an EMA to meet its task demands as compared to its as-built condition), etc. [Hvass and Tesar, 2004]. With multiple sensors monitoring different characteristics, it may be possible to accurately pinpoint the cause of a defect using the available sensor information (for instance, excessive vibrations may be traced to a particular defective bearing [Shakkari and Tesar, 2007]). The sensor values are also used to constantly update the maps to reflect the latest EMA health condition for subsequent cycles of the CBM algorithm.

1.3. DECISION MAKING IN INTELLIGENT EMAS

In many intelligent systems, the preferred approach is to minimize or eliminate human involvement in decision-making for greater autonomy[14]. But automated decision making has not always improved system performance, either due to problems in interpretation of their results or due to unanticipated interactions between sub-systems leading to erroneous verdicts [Parasuraman, et. al., 2007].

Resource allocation based on task requirements is difficult to program into an automated decision-making system, more so if it operates in an uncertain, dynamic environment. With a fully autonomous (rule-based) decision-making arrangement, if the operating conditions vary drastically from the norm (or the data on which the decision-making algorithm was trained on), the system may not respond satisfactorily. Reliance on complete automation for decision making has also been directly attributed to a lower level of operator awareness, sometimes leading to safety critical situations (e.g. in avionics [Qureshi and Urlings, 1999]). [Hunn, 2008] states that the out-of-loop

[14] Primarily for reasons of safety, the ability to supersede automated decision-making and assume charge of crucial operations in case of unexpected events or sudden changes in task requirements is still considered a human responsibility. Even in highly autonomous systems (like mobile robots) human operators are involved in the system operation to a large extent by defining rules/operational constraints required to achieve the mission objectives, interpreting data streamed from on-board sensors, diagnosing issues, etc.

performance leaves the human operators of such systems handicapped in their ability to contribute their expertise to make the system perform optimally or take over operations when needed. [Parasuraman, et. al., 2007] describe the common pitfalls of completely automated decision-making as over-reliance, skill degradation, and reduced situation awareness, especially in emergencies. [Parasuraman, et. al., 2007] also highlight studies by [Crocoll and Coury, 1990] and [Rovira et al., 2007] (for aircraft identification and battlefield engagement respectively) involving automation support for decision-making that was not perfectly reliable and provided either status information or a task recommendation. It was found that the end cost was greater when end-users acted on a flawed verdict automatically provided by the decision-making system (decision automation) than when they were provided only status information from which they derived their own decision (information automation).

The above findings imply that in dynamic/vaguely defined scenarios, critical judgment based on present context rather than pre-programmed options is vital for successful decisions with the operator also actively involved in the decision-making process [Tesar, 2010 (4)]. Human-in-the-loop approaches, with human and machine contributions toward a complex end goal defined by the human operator, are therefore being explored in many domains to utilize the best capabilities inherent in both [Hunn, 2008][Kaupp, 2007]. With enough computing resources, it is possible to rapidly analyze vast amounts of low-level raw data to derive higher level information or features using suitable algorithms. But with uncertain or incomplete information, completely automated decision-making algorithms are generally unable to infer enough about the system status and may even arrive at erroneous decisions. Conversely, humans can naturally infer the context of any situation, assimilate partial/uncertain knowledge, prioritize new information or eliminate redundant information and make decisions adaptively using a combination of past experience and available facts[15]. Such an ability is often unavailable to machine systems, where decision-making is more deterministic. A human in the loop

[15] This ability to infer patterns in data or imagery that intuitively seem to fit without detailed analysis is referred to as 'Gestalt perception' [Hunn, 2008].

approach to decision-making thus provides an improved ability to understand/interpret system behavior due to the human ability to extrapolate *a priori* knowledge of what the system can and is likely to do, based on experience and combine it with measured data. [Parasuraman et. al. 2000] suggest that in complex systems, early-stage functions like data acquisition, processing, low-level control functions, etc. may be automated with great benefits. But for higher-level (or high-risk) operational decisions, unless the automated decision-making system can be certified to be fully reliable under all possible scenarios, the human operator should not only be involved in the constituent processes of the decision-making cycle, but also make the final decision. The arrangement ensures that the operator is not over-burdened with a deluge of raw data but still retains the flexibility to integrate different sources of information before arriving at a final decision.

It is widely accepted that a synergy of human and machine intelligence can provide an opportunity to enhance the capabilities of today's systems, and increase their robustness and reliability [Pavlin, 2007] [Kaupp, 2007] [Bruemmer, et. al., 2005] [Fong et. al. 2002] [Ross, 1998]. A pertinent example of human-in-the-loop decision-making can be found in mobile robots deployed in combat zones or urban search and rescue which rely on human-robot (intelligent system) interaction rather than complete autonomy for greater flexibility in responding to limitations imposed by system design or uncertain operational environments [Lewis, et. al., 2007]. The UT-RRG philosophy of EMA intelligence (Figure 1-4) is modeled on similar lines in that it is imparted as a means to realize and expand the choices of a human arbiter for maximum operational flexibility at the EMA level [Tesar, 2009][16]. While lower-level data acquisition and control functions are automated, human operators are responsible for supervisory control, providing the final decision for EMA operation. This is done by visually providing information regarding the EMA parameters to the user in the form of decision surfaces (performance maps/envelopes) and norms derived from these surfaces. These are used in combination with user-defined performance criteria and sensor-inferred EMA

[16] Since, an EMA forms a basic building block of higher-level systems (Section 1.1), this automatically helps expand the operational choices available to the operator to improve the overall system performance.

characteristics (the decisions are made by the operator based on residuals between the maps and the actual sensor-inferred values) to satisfy various task demands. The feasibility of using such an interactive approach to decision-making has been verified in [Ashok and Tesar, 2007][Hvass and Tesar, 2004] and [Yoo and Tesar, 2004].

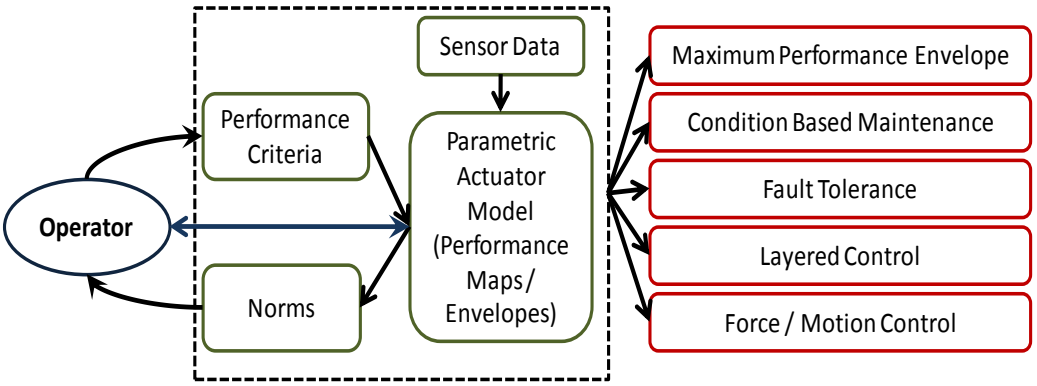


Figure 1-4 : Decision making in Intelligent Electromechanical Actuators

1.4. RESEARCH PROBLEM

The quality of human-in-the-loop decisions will only be as good as the information available to the decision maker. Reliable and accurate sensor data from an extensive set of sensors forms the basis for generating all the maps/envelopes used in the decision making process as well as in real-time evaluation of the EMA capabilities. Some factors which affect the quality, availability and the use of sensor data and their impact on the operator/decision-maker are as follows:

- In practice, sensors usually do not operate in exact accordance with their theoretical models. All sensor readings have an element of uncertainty due to inherent sensor physics or external factors which add noise to the signals or cause some form of sensor malfunction[17]. This in turn, is reflected in the form of

[17] Noise is manifested as high frequency random fluctuations from the true sensor reading. It is commonly observed when sensors are located close to components like switching power supplies. Noise can also be added to a signal at various stages: from the measurand itself during the sensing process; "picked up" in the cabling, digitization sampling noise, etc. The aim is always to maximize the Signal to Noise Ratio (SNR).

increased uncertainty in the decision surfaces used for EMA control[18]. Concepts such as CBM rely heavily on the differencing of such decision surfaces [Hvass and Tesar, 2004]. To reduce false alarms from these algorithms, it is essential to ensure that the resultant difference is accurately indicative of performance degradation, after accounting for the inherent uncertainties.

- Often there is a need to infer quantities that are not directly measurable (e.g. efficiency). This is usually achieved via analytical/functional models that incorporate available sensor data. The accuracy of the inferred value depends on the quality of the models used as well as the quality of measurands used as model inputs. In such cases, the resultant uncertainty needs to be propagated to the higher levels of the decision-making hierarchy (an EMA controller or the human who controls the final decision-making) in order to maintain awareness of this uncertainty at the system level.
- A sensor with mechanical/moving components experiences wear and degradation during its operational life. Solid state sensors are also subject to extreme conditions (e.g. high temperature, shock/vibrations) when integrated into the confined spaces within an EMA. Failure in sensor harnesses and connectors are also quite common [Puls and Tesar, 1994]. In such scenarios, an arrangement using single sensors (absence of any hardware redundancy) is not very robust since any one sensor failure may result in complete system failure. If the outputs from faulty (or failed) sensors are used in updating the available performance maps/envelopes outputs, it may cause false alarms from CBM algorithms which may result in the EMA being replaced earlier than necessary in its lifecycle. In feedback control, faulty sensors may cause disastrous results as the system may become unstable, or may provide poor regulation in precision applications [Xu and Kwan, 2003]. Failure to identify an 'abnormal' sensor and take appropriate corrective action could result in expensive and unnecessary system shutdowns or,

[18] The parametric models or maps may have inherent modeling uncertainty arising from how they are generated (experimentally or analytically) [Ashok and Tesar, 2007].

worse, accidents that endanger both system and personnel. Typically, it may not be warranted to replace the whole EMA in case of loss of sensor information. It is usually not practical to replace a failed sensor(s) when the EMA is in service. Hence, some safeguard is needed to detect signs of any form of sensor failure or degradation and provide advanced warning.

- To balance choices like sensor geometry (for compact integration), performance (accuracy, resolution, etc.), power and signal processing requirements, cost, etc. compromises are often made in the selection of sensors for an EMA application. Sensor ratings are usually selected to conform to nominal EMA operating conditions since these conditions prevail for a majority of its operational life. Hence, all sensors do not perform equally well over the entire operating period of the EMA and a significant deviation from desired performance is often observed especially in regions close to the operational envelope of the EMA. Hence, some form of information redundancy (using different sources in different operating regimes for the same measurands) may be needed to achieve better data fidelity.
- Individual sensors can only present a partial appraisal of the operational capability of an EMA or its environment. Therefore each sensor needs to be used concurrently with other available sensors for a more realistic assessment of the present EMA operating conditions.

The objective of this work is not to try and improve the design of off-the-shelf sensors (which often have tremendous capabilities), but to make the best of the existing capabilities of these sensors while addressing some of the shortcomings discussed above through EMA operational software (Appendix C). Post-acquisition, the first step is to filter the raw sensor data to separate useful information from extraneous noise components. This data needs to be further bolstered via computational tools to ensure its ‘correctness’ and used in concert with information from other sensors to provide information of greater value, which can be used by the system operator for more reliable control and diagnosis.

1.5. RESEARCH OBJECTIVES

The benefits of EMA intelligence cannot be achieved if input data to the decision-making process (Figure 1-4) is of questionable quality. But operational decisions still have to be made in spite of imprecise and uncertain sensor data. Some core issues that have been identified to define the research objectives for this work are as follows:

I. With any unwarranted deviations in sensor readings from their expected behavior (based on past operational history or available calibration data), the challenge is to determine if the variation indicates a change in the states of the monitored system or if it is the result of one or more malfunctioning sensors. However, it is often difficult to detect abnormal sensor behavior as it can be masked by normal system changes (for example, a change in the EMA operating speed)[19]. In most approaches to sensor fault detection, the underlying assumption is that the monitored system itself is operating correctly with no existing or incipient faults. Conversely, with most system level fault detection approaches, the principal assumption used is that all the sensor readings are perfectly valid and any deviation in readings is the result of a fault in one or more sensors. Thus, the first research objective is to:

   Create a unified framework that can detect any system faults and sensor faults concurrently, independent of any assumptions regarding the health/condition of either the sensors or the monitored system.

II. When readings from individual sensors are used in isolation, there is no mechanism to evaluate their correctness. But, by correlating all the measurands using a suitable framework, it may be possible to detect the presence of any errors in available data. The first step is to use the information redundancy enabled by such correlation, to generate multiple values for each measurand that can then be compared against real-time measured data. In the second step, based on the

[19] An abrupt sensor failure, caused by power failure, loose/corroded contacts, or data acquisition system faults, etc. is relatively easy to detect (for instance, with simple limit checking). However subtle faults like drift (caused by deterioration of the sensing element) or cross sensitivity are more difficult to detect.

closeness of agreement of the predicted and measured values[20], a measure of belief may be assigned to the readings from each sensor. By tracking such measures over time for all the sensors, it may be possible to distinguish a faulty sensor from a faulty system[21].

Multiple values may be estimated for a variable of interest via analytical or functional redundancy. With analytical redundancy, the values are synthesized from direct mathematical models[22], relating a measurand to its dependent variables (e.g. *V=IR*). However, such relationships are unlikely to exist for all combinations of measurands. Functional redundancy is a more general approach in which the variable is estimated via readings from non-redundant sensors that are not measuring it directly, but additional variables that are related to it by some physical process. Usually, nonlinear models derived from empirical data (such as performance maps, look-up tables or regression equations [Janardhan and Tesar, 2008] [Ashok and Tesar, 2009]) are used for this purpose. In doing so, data from multiple sensors must be brought together coherently to avoid propagation of conflicting information through the control hierarchy. Thus, the second research objective is to:

Use an empirical data-driven approach to functionally correlate the variables of interest in an EMA in a physically meaningful manner.

III. Any modeling framework that is used to associate the variables of interest must be able to numerically quantify the inherent uncertainties that exist in the sensor readings (due to limitations on linearity, accuracy, resolution, etc. dictated by the physical principles unique to each specific sensor) as well as in the presumed relations among the variables (due to the inability to account for all the factors that may influence a specific variable or due to simplifying assumptions made

[20] Such a comparison may be carried out based on user-defined thresholds, statistical measures, etc.

[21] In the simplest scenario, if the measure of belief for a single sensor seems to be out of line compared to others then it is likely that the particular sensor may be faulty whereas variation in the measures associated with multiple sensors may either be indicative of multiple sensor faults or an incipient system level fault.

[22] Typically linear for ease of computation.

during experimentation, data fitting, etc.)[23]. There is also the need to have formalized procedures (that are also mathematically tractable) for manipulating and propagating these uncertainties across different levels of the resulting framework as well as in updating the resultant model as and when new information becomes available (primarily for CBM algorithms). Such techniques also need to be scalable to address system models with a large number of variables of interest and the different combinations in which the variables can be potentially related. Thus, the third research objective is to:

Utilize well-defined mathematical principles to enable quantification, combination and propagation of uncertainties resulting from different facets of the system operational decision making process, in a computationally tractable and scalable manner.

IV. Often, it may not be possible to represent all the relevant domain knowledge via functional models. Hence, it is essential to implement mechanisms that can allow human system operators to utilize their experience and knowledge of the EMA behavior under different operating conditions to carry out tasks such as dynamically selecting suitable sensors to act as data sources for updating the available performance maps (especially when the quality of information from one or more sensors seems to be deteriorating), modify/ select appropriate control strategies to account for any deterioration in the condition of any of the EMA components (bearings, motor windings, etc.), identify suitable sensors that may be used to provide alternative information pathways for failed sensors, etc. To enable such decisions, there is a need to provide an interface that maximizes the operator's comprehension of what is actually happening within the subsystems. A quick and intuitive way to achieve this is to present all the component factors required for decision making in a visual and interactive manner [Ashok and Tesar, 2008][Hunn, 2008][Ruspini, et. al, 2005]. A proven application of such an

[23] Note that these uncertainties will be automatically manifested in the available empirical data that is used to generate the performance maps

approach is the use of head-up displays for aircraft. In the present context, in addition to a visual display of the appropriate performance maps and envelopes (as described in [Ashok and Tesar, 2007] and [Yun and Tesar, 2007]), additional resources that can assist an EMA operator (or even a domain expert) in making operational decisions are easy to interpret visual representations of the influences among different physical variables in the system as well as the quantitative indicators that can provide a preliminary assessment of the health/condition of each component of an EMA and every sensor integrated into the EMA (a detailed prognosis for these components as well as the overall EMA health could be obtained from more comprehensive CBM algorithms as proposed by [Hvass and Tesar, 2004] [Shakkari and Tesar, 2007], etc.). Thus, the fourth research objective is to:

Provide a visual representation of the relationships among the different physical variables to complement performance maps as well as preliminary quantitative indicators for the condition of different system components and sensors in order to enable a human-in-the-loop approach to operational decision making for optimal allocation of the available resources.

1.6. REPORT OUTLINE

This chapter outlines the philosophy of intelligence in mechanical systems like EMAs, emphasizing the need for a human-directed decision making paradigm utilizing a combination of decision surfaces (performance maps and envelopes), user-defined performance criteria and real-time evaluation of all the physical variables in the system using a suite of multiple, heterogeneous sensors. Given the central importance of sensor data as the foundation for such intelligence, various issues in ensuring the availability and quality of such data were identified in Section 1.4 and the research objectives for this work were listed in Section 1.5.

Chapter 2 provides the argument for using Bayesian networks as the method of choice to graphically model the complex associations between the numerous physical

variables as well as to provide a formal method to account for uncertainties (using probability theory) in a system like an intelligent EMA. The chapter provides a brief review of the fundamentals of Bayesian networks, and concludes with the description of three popular probabilistic inferencing algorithms that can be used for forward/inverse propagation of uncertainties across the model.

Based on some of the concepts discussed in Chapter 2, Chapter 3 describes the development of a comprehensive Bayesian network for a representative EMA test bed comprised of a four phase Switched Reluctance Motor (SRM) with other auxiliary components and monitored using an array of sensors (the details of the testbed are provided in Appendix B). The overall network topology is developed by linking various smaller models representing the causal relationships among diverse physical facets of motor operation such as torque generation, motor dynamics, generation of acoustic noise, motor thermal characteristics, etc.

Chapter 4 delves into the need for a criteria-based sensor management framework to help decide on and optimally utilize sensing resources in any complex system to their fullest potential. The chapter discusses various criteria that may be used to decide the physical embodiment of sensors in the system as well as in developing its corresponding Bayesian network representation. The chapter also describes additional criteria that may be used by the system operator to prioritize the available sensor and computational resources, in order to improve the operational use of the network thus developed.

Chapter 5 presents the development of a novel algorithm referred to as the 'Sensor and Process Fault Detection and Isolation (SPFDI) algorithm. In case of any unexpected variations in sensor readings, the proposed algorithm utilizes the available Bayesian network representation of a complex system like an EMA and the real-time information obtained from a suite of multiple sensors to help distinguish between sensor and process/system faults in tandem (without the need for simplifying assumptions regarding the condition of either the sensors or the overall system). The algorithm also provides a quantitative belief value representing the health of each sensor or sub-component that can be used by the system operator during the decision-making process.

Chapter 6 then discusses four broad application classes (each consisting of a complex system equipped with multiple sensors and utilizing a human-in-the-loop approach to decision making) with different objectives (viz. human safety, system availability, cost of system failure, and performance maximization) where the SPFDI algorithm may be applied with beneficial results, Three representative examples are presented for each class of applications, with an in-depth description of a selected example in each class,

Chapter 7 provides a summary of the work presented in this report and also outlines future avenues of research.

Chapter 2. Bayesian Networks

2.1. INTRODUCTION

The objective of this chapter is to provide the motivation and the necessary mathematical framework for the use of Bayesian networks as a viable method for modeling intelligent EMAs, capable of handling both nonlinearities and uncertainties associated with the different operational parameters of an intelligent EMA. Section 2.2 briefly outlines the desirable characteristics in modeling techniques that are needed to describe a system, and also provides a brief discussion on various formalisms to handle uncertainty. Section 2.3 describes various concepts and definitions in probability theory, including the Bayes' theorem which forms the core of the Bayesian reasoning approach. Section 2.4 formally defines Bayesian networks and delves into its different properties. This is followed by a description of different algorithms that may be used to propagate, combine and update the uncertainties (represented using a probabilistic approach) associated with the different variables of interest.

2.2. MODELING OF INTELLIGENT EMAS

As an initial step to achieve the objectives listed in Chapter 1, it is necessary to model the interaction between the various actuator parameters with a view to provide functional redundancy for all the parameters of interest. The goal is thus to create a comprehensive EMA model based on the parameters of interest which can be directly measured from sensors integrated into the system (or accurately inferred using the available measurands). Some criteria to evaluate a good system modeling technique may be adopted from the list provided by [Zhang and Bivens, 2007][1] as follows:

- The model obtained as a result of the adopted methodology should accurately represent the system (or available data) and should have adequate flexibility in rapidly changing environments where the model has to be frequently reconstructed

[1] Although the focus of the authors work is in computing systems, the requirements may be considered to be generally applicable to other types of systems too

- The evaluation speed of the resulting model should be quick, in order to enable it to be used for system response predictions with high intensity
- The modeling technique should accord the model designer the ability to incorporate existing domain knowledge. Such knowledge may be in the form of expert experience or hard empirical data. The objective is to exploit this knowledge to enhance model performance using readily available information during system operation.
- Any model that is created using such a technique must have easy interpretability, in order to enhance human understanding of the model as well as the confidence to use the model during system operation.
- The technique should allow extension of the use of the resulting model to capturing different behaviors, thereby increasing its capability to accurately represent the system.
- The technique should support multi-way model evaluations, for example cause to effect as well as effect to cause inference or inferring data not available because of restrictions imposed by system design.
- The modeling technique should allow for incorporation of different types of uncertainty during analysis.

The ability to account for uncertainty is particularly important. Conventional engineering analysis tends to be deterministic, based on single-point values of variables of interest. However, this is never true in reality because every variable has some degree of uncertainty associated with it. [Korb and Nicholson, 2004] describe the three distinct forms of uncertainty an intelligent system has to be able to handle in general- ignorance (caused by limits to knowledge), indeterminism (arising from inherent randomness in physical quantities), and vagueness. Some of these uncertainties are inherently introduced as a result of any modeling process. For instance, mathematical equations that are used to characterize a physical phenomenon are usually arrived at empirically and are an approximation of the actual phenomenon at best. These equations are often further simplified for further use, using certain assumptions, for computation reasons. In

addition, the sensors used to measure physical variables are not purely deterministic and add an additional element of uncertainty. Uncertainty in sensor data may result from many sources like numerical errors in signal processing algorithms, A/D conversion, etc. Hence, quantifying the accounting for uncertainty in a variable of interest is just as important as the exact value of the variable itself.

Uncertainty in sensor data may be characterized using different methods as described by [Brooks and Iyengar, 1998][2]:

- *Explicit accuracy bounds using statistical techniques:* Data values are assumed to follow a known statistical distribution (a Gaussian distribution is commonly used since it has many favorable attributes both from mathematical and computational perspectives) centered on the actual sensor reading as the mean value. The accuracy or certainty bounds for a given sensor are then derived statistically by finding the distribution of the signals generated under normal operating conditions.
- *Probabilistic bounds:* Each data value is assigned a probability value ranging from 0 to 1, and suitable mathematical techniques may be used to combine the probabilities to produce a single (un)certainty value.
- *Belief functions:* The Dempster-Shafer theory of belief functions computes the probability that the available information/evidence supports a particular proposition rather than computing the probability of the proposition itself [Russell and Norvig, 1995]. A belief function is usually represented as a probability interval instead of a single value.
- *Fuzzy logic:* Each data value is viewed as a membership function of a fuzzy set. Rules are then defined for quantifying a sensor reading using the values of these membership functions.

[2] [Naish and Croft, 1999] also define the concept of uncertainty ellipsoid (based upon the normal distribution), utilizing a geometric ellipsoid data representation to quantify uncertainty. The center of the ellipsoid is the mean of the measurement and its boundary represents a distance of one standard deviation from the mean. A beneficial feature of this model is that it can represent data of any dimension using the same mathematical framework.

The work of [Walley, 1996] provides a detailed comparison of the different uncertainty handling approaches mentioned above on the basis of six criteria and is also summarized in [Ashok and Tesar, 2007] as shown in Table 2-1.

Table 2-1: Comparing Uncertainty Measures

Criterion	**Probability Theory**	**Dempster Shafer Belief Functions**	**Fuzzy Logic Possibility Measures**
Interpretation (For use as basis of action)	Simple behavioral interpretation	Interpretation is not very well defined.	
Imprecision (Ability to model ignorance, partial/ incomplete or conflicting information, and imprecise assessments of uncertainty?)	Unable to model ignorance or imprecise judgments of uncertainty	Can model ignorance and limited evidence	Can model imprecise or partial information
Calculus (To combine measures of uncertainty, update them based on new information, determine uncertainties in associated variables for decision making, etc.)	Well developed mathematical formalism	Simple combination rules but not well developed	Simple but arbitrary rules
Consistency (Availability of methods for checking consistency of all uncertainty estimations and assumptions used)	The calculus is developed based on principles of consistency	Not very reliable	Lack methods for checking model consistency
Assessment (Practicality in terms of user interpretation of available uncertainty assessments)	Intuitive numerical estimates	Not straightforward.	Requires translation of natural language judgments
Computation (Computational complexity for the system to derive inferences and conclusions from the assessments)	Complexity depends on model structure	Computationally efficient methods have been developed.	Nonlinear programming involving complex computations

(Reproduced from Ashok and Tesar, 2007)

Each of the above methods has been successfully applied in different applications and each is more suitable to a particular domain/objective than others. For instance, belief functions are more suitable in modeling ignorance whereas fuzzy logic is more suitable in modeling vagueness [Russell and Norvig, 1995]. For the present work, qualitative judgments about vagueness or ignorance are not major issues (but accounting for variations in sensor signals is) since the performance maps based decision making methodology for intelligent EMAs is based on extensive experimental data that is obtained by rigorous experimentation as presented in [Janardhan and Tesar, 2008]. For

the same reason i.e. the availability of extensive empirical data, the probabilistic approach has been deemed to be the most suitable approach to quantify uncertainty [Ashok and Tesar, 2007] [Das, 2004]. In dynamic environments where the system model has to be updated or reconstructed regularly, [Zhang and Bivens, 2007] suggest that a probabilistic model is more suited for capturing the uncertainties. In addition, the use of probability allows encapsulation of a large set of circumstances in an approximate but very concise manner [Russel and Norvig, 1995]. Both these factors are important from the point of view of intelligent EMAs where the performance maps need to be periodically updated (locally as well as over the entire range) to accurately reflect the extant conditions. Once the uncertainty is quantified, it is also essential to determine how the modeling technique can enable these values to be propagated and combined in order to examine their effects on different parts of the model (for instance, if the model for a particular system is of the form $A = f(B)$, the goal is to determine how uncertainty in the value of B affects the uncertainty in the estimate of A).

[Ashok and Tesar, 2007] also provide a detailed comparison of some commonly followed approaches for modeling intelligent EMAs such as state-space modeling, neural networks, etc. In the state-space modeling approach, the models are physics based (which provide a sound analytical basis along with well-defined model interpretation) and experiments are conducted to find the constants in the model after it has been finalized. But such models are typically not robust enough to handle uncertainty in data unless they have been explicitly included in the model when it is created. Also, multi-way (forward-inverse) model evaluation, uncertainty propagation, etc. is not always easily handled in this approach nor is it always physically relevant because of the simplifying assumptions (e.g. linearization) made during the modeling process. On the other hand, approaches like neural networks are data-based and have the ability to learn complex patterns directly from actual (noisy) data [Berchialla, et. al. 2007]. They tend to perform well in many multivariate, nonlinear applications (for e.g. as described in [Correa, et. al., 2009][Zheng, et. al, 1999], etc.) but their reasoning process is inaccessible to human understanding (they are often referred to as ‘black-box’ models). The resultant model is implicit in the

network structure (which is not necessarily based on physics or engineering principles, and therefore may lack a strong analytical foundation) and no explicit knowledge representation is available in a form that can be easily understood by the human system operator [Zheng et. al. 1999]. While such approaches permit the solving of many problems, they are typically structured to solve one particular problem. Furthermore, due to gradient-based optimization processes involved, possible data over-fitting may significantly deteriorate the robustness of neural network based schemes (i.e. the model may not perform equally well over the entire operating range of the system).

To address the above issues, there is a need for a graphical and intuitive model presentation (based on physical principles) and can not only assist users in quickly grasping the model, but is also data-driven (allowing for propagation of uncertainties) with a solid mathematical foundation (facilitating forward and inverse inference of any variable), thereby increasing the user confidence in its correctness [Zhang and Bivens, 2007]. These needs are best served by using a Bayesian network framework to model intelligent EMAs in terms of its measurands. Such an approach also complements the overall EMA decision-making methodology developed in [Ashok and Tesar, 2007].

2.3. PROBABILITY CONCEPTS

The following section provides an overview of some essential probability concepts that form the basis for Bayesian networks. Many of the definitions provided in this section are drawn from [Pearl, 1988], [Neapolitan, 2003], [Korb and Nicholson, 2004], and [Darwiche, 2009]

Definition 2.1: Consider a sample space S consisting of n distinct events i.e. $S= \{e_1, e_2,\ldots e_n\}$. A function P is a probability function on the subsets of S, if it assigns a real number $P(E)$to every event $E \in S$ and has the following properties[Neapolitan, 2003]:

1. $0 \leq P(E) \leq 1$ and $P(S) = P(e_1)+ P(e_2)+\ldots.. P(e_n)=1$
2. For disjoint subsets e_{i_k} of non-elementary events $E=\{e_{i_1}, e_{i_2}, \ldots e_{i_k}\}$

$$P(E) = P(e_{i_1})+ P(e_{i_2})+\ldots+ P(e_{i_k})$$

The pair (S, P) is then known as a probability space[3]. The term probability is thus used to quantify the level of belief regarding the occurrence of an event. In other words, it can be used as a measure of uncertainty and quantify it.

Definition 2.2: If A and B are any two events such that $P(B) \neq 0$, then conditional probability (i.e. given that the event B occurred what is the probability of the event A happening) is defined as follows [Neapolitan, 2003]:

$$P(A \mid B) = P(A \cap B)/P(B) \qquad \text{Eq. (2-1)}$$

Definition 2.3: Two events A and B are said to be independent (represented by '$\perp$') if $P(A|\ B) = P(A)$ for $P(A) \neq 0$ and $P(B) \neq 0$ [Neapolitan, 2003]. The independence relation is symmetric. In addition, $P(A \cap B)=P(A)P(B)$.

Definition 2.4: If A is independent of B only with the knowledge of a third event (or a set of events)C, where $P(C) \neq 0$, then such a situation is referred to as conditional independence and is represented as ($A \perp B \mid C$)[4]. In this case, $P(A|\ B \cap C)= P(A|\ C)$ for $P(A|\ C) \neq 0$ and $P(B|\ C) \neq 0$ [Neapolitan, 2003]

Definition 2.4: The total probability for any event A, given a set of n mutually exclusive and exhaustive events $B_1, B_2, \ldots. B_n$ that constitute the space S (i.e. for any $i \neq j$, $B_i \cap B_j = \Phi$ and $B_1 \cup B_2 \cup \ldots \cup B_n = S$) is defined as [Neapolitan, 2003]:

$$P(A) = \sum_{i=1}^{n} P(A|B_i)P(B_i) \qquad \text{Eq.(2-2)}$$

Definition 2.5: The concepts discussed above are equally valid when specified in terms of *random variables* instead of events. Analogous to events, a random variable can be defined as one which takes on a unique value corresponding to every outcome in the sample space of interest [Neapolitan, 2003]. The individual values which such a variable

[3] The function P is also known as a probability distribution when used in the context of random variables
[4]This is sometimes represented as $I(A, B|C)$. When C is an empty set, the relation reduces to the marginal independence relation.

can assume are referred to as its '*states*' and the combined set of states is referred to as the variable's state space [Korb and Nicholson, 2004]. Random variables may be continuous or discrete. Continuous random variables may assume any value over the entire range of real numbers while discrete random variables have a finite set of states.

For the present work, all the variables are assumed to be in the discrete domain[5]. An example random variable in the context of the present work would be the reading from a sensor corresponding to a particular variable of interest in the system, like torque or speed. Even though many real-world phenomena (like temperature) can be considered to be continuous in reality, when such variables are measured using sensors, any information regarding their variation is available in the form of discrete steps due to the signal processing associated with the sensors and their auxiliary data acquisition equipment. In other words, continuous values are approximated by using discrete values. For instance, if the motion of a robot joint is measured using a position sensor, and the absolute joint position (assuming the starting position represents 0^o) is represented by the random variable J_P, then J_P may take discrete values as $J_P = 0.5^o$, $J_P = 1.0^o$ and so on as indicated by the sensor reading. The greater the resolution of the sensor used, the more the number of discrete states that can be measured. The granularity of discretization (i.e. number of states) for the variables is typically dictated by the application requirements.

Introducing some notations, the discrete random variables are denoted by capital letters as A, B, C, etc. The distinct values assumed by each of these variables are denoted by the lower case letters as a, b, c, etc. The probability that a variable A assumes the value a is denoted as $P(A=a)$ or simply $P(a)$. The function $P(A=a)$ for different values a is then referred to as the probability distribution of the random variable A. The conditional probability that a variable A assumes a value of a when the variable B assumes a value of b is similarly represented as $P(A=a|B=b)$ or simply $P(a|b)$. The joint

[5] Primarily because a large number of the algorithms that have been developed in the realm of Bayesian networks (Section 2.5) are focused towards dealing with discrete variables There are also algorithms that deal with continuous variables (Gaussian) or mixed networks with both continuous and discrete variables as described in [Sudderth, et. al, 2003], [Neapolitan, 2003], [Darwiche, 2009], [Cobb and Shenoy, 2004], [Chang and Sun, 2003] and many others.

probability distribution of A and B is denoted as $P(A=a \cap B=b) = P(A=a, B=b)$ or $P(a,b)$. The marginal probability distribution of A is then defined as $P(A=a) = \sum_b P(A = a, B = b)$ where $\sum_b$ indicates that the summation encompasses all the values in the state space of B. A set of random variables V, with a joint probability distribution defined over all these variables is referred to as a *probabilistic model* [Pearl, 1988].

2.3.1. Bayes' Theorem

The probability concepts discussed thus far now enable the definition of the Bayes' rule or Bayes' theorem. This provides a mechanism by which conditional probabilities of variables of interest may be determined using known or available probabilities when new information becomes available. The theorem can be stated as:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \qquad \text{Eq.(2-3)}$$

The probability of a hypothesis H conditioned on some evidence E is computed by multiplying the probability of its likelihood $P(E|H)$ with the prior probability of the hypothesis $P(H)$ and ultimately normalizing by $P(E)$ i.e. the probability of observing the evidence. The normalization ensures that the conditional probability of all such hypotheses sum to 1. The quantity $P(H|E)$ is referred to as the posterior probability of the hypothesis after taking into account the evidence and is often termed as 'belief' [Korb and Nicholson, 2004]). The process of updating the beliefs (or propagating the uncertainties) is referred to as 'inferencing' (Section 2.5). The goal is thus to use the states of those variables which are available or can be observed easily in order to determine those of other variables which are harder to observe directly or are unavailable. The above equation may be generalized to a set of n mutually exclusive and exhaustive hypotheses $\{H_1, H_2, \ldots H_n\}$ such that $P(H_i) \neq 0$ for $i=1,2,\ldots n$,

$$P(H_i|E) = \frac{P(E|H_i)P(H_i)}{\sum_{i=1}^{n} P(E|H_i)\,P(H_i)} \qquad \text{Eq.(2-4)}$$

In the context of performance maps, the Bayesian approach essentially provides a way to modify the system models represented in those maps (H) when new data becomes available from the sensors (E) that are used in monitoring the system.

Consider two random variables *A* and *B* each with two states such that $P(B=b_1) = 0.1$, $P(B=b_2)=0.9$, $P(A=a_1|B=b1)=0.6$, $P(A=a_1|B=b_2)=0.8$. Now in order to calculate P $(B=b_1|a=a_1)$, Bayes' rule is applied as follows

$$P(b_1|a_1) = \frac{P(a_1|b_1)P(b_1)}{P(a_1|b_1)P(b_1) + P(a_1|b_2)P(b_2)} \quad \text{Eq.(2-5)}$$

The process of Bayesian inferencing is simple when there are only two variables involved, each with a finite number of states as shown above, since it is easy to model the joint probability distributions (there is a need to specify only 2^2 probability values that cover all possible combinations of the values of *A* and *B*). However, in any complex system, there may be numerous operational parameters, each represented by a random variable. As the number of variables increases, the specification of the joint distributions (eliciting their probability values) and calculations of the normalizing constants in the above equations becomes computationally very complex. For instance, in a system consisting of *n* variables with *m* states each, n^m probability values would have to be estimated (a daunting task if the values are obtained by expert opinion and even if it is obtained by experimentation) Bayesian networks have been proposed as a possible solution to the above issues [Darwiche, 2009] due to their ability to make the problem more tractable by exploit the inherent dependence and independence relationships that exist between the variables in the domain of interest.

2.4. BAYESIAN NETWORKS

Bayesian belief networks are referred to using many different names- belief networks, causal networks, probabilistic networks, etc. The concepts used are derived from fields of graph theory, probability theory, and statistics. They have gained popularity in recent years as a method for both modeling of complex problems (incorporating uncertain knowledge) as well as for decision making/ performing reasoning under uncertainty [Pearl, 1988]. These networks have been used for modeling multinomial data (both discrete and continuous). In essence, a Bayesian network represents the interconnectedness between the different random variables that represent the parameters of interest in a given domain. The graphical framework of Bayesian

networks not only provides an intuitive understanding of the domain being modeled but also allows for a compact representation of multivariate probability distributions (by representing the joint probability as product of local distributions). Such models have been used in a wide variety of application areas such as medical diagnosis [Spiegelhalter, 1989][Acid, et.al., 2004], fault detection [Lampis and Andrews, 2009][Li et. al., 2007][Ibarguengoytia, et. al. 2005], gene analysis [Freidman et. al. 1999], situation assessment in military applications [Bladon, et. al, 2006], etc. to name a few.

2.4.1. Definition

A Bayesian network is a graphical structure that consists of a set of nodes that represent the random variables $X = \{X_1, X_2, \ldots X_n\}$ pertinent to the physical domain of interest. The random variables may be discrete or continuous. These nodes are connected in pairs by a set of directed arcs (or links or edges) which explicitly represent the probabilistic dependencies between the variables. The lack of an arc between two nodes explicitly represents their independence. The direction of these arcs is such that no cycles (a directed path from a node to itself [Neapolitan, 2003]) are formed. The structure is therefore referred to as a Directed Acyclic Graph (*DAG*).

If a node A and a node B are connected by a directed arc as $A \rightarrow B$, then B is said to be conditionally dependent on A. If the directed edges indicate direct causal influences or dependencies between the variables then the network is also known as a Bayesian causal network[6]. A is said to be the parent node of B and B is its child. A is said to be an ancestor of B if there is a path formed by such links from A to B and B is said to be its descendant in this case. Any other node C which is not a descendant of A is said to be a non-descendent. A node which has no parents but only children is referred to as a root node. Conversely, a node which has no children is referred to as a leaf node. Any non-root or non-leaf node is referred to as an intermediate node [Korb and Nicholson, 2004]. In a

[6] The distinction between a purely causal network and a Bayesian causal network is provided in [Nadkarni and Shenoy, 2001] [Ashok and Tesar, 2007]. Causal networks are directed maps that are primarily used to represent domain knowledge in terms of cause-effect relations but they need to be conditioned into Bayesian networks to allow propagation of uncertainties. Conversely every relation represented in a Bayesian network need not necessarily represent causality.

causal network, the parent nodes in the network can be considered to directly influence their children nodes, with the cause-effect relation represented by the link starting from the influencing or the parent node and terminating at the child node.

The structure of a Bayesian network may assume different forms as shown in Figure 2-1. In case of a tree structure (Figure 2-1(a)), any node in the network may have at most a single parent. In case of a singly connected or a polytree structure (Figure 2-1(b)), a node may have multiple parents but there is only one path between any two nodes in the network. In case of a multiply connected structure (Figure 2-1(c)), a node can have multiple parents and there may be more than one path between any pair of nodes in the network. This is the most general type of network. In general, multiply connected networks are more useful when modeling real-world systems but as will be seen later, probability propagation in such structures is more complicated as compared to trees or singly-connected networks. The strength of the relationships (dependencies) between the nodes (and the inherent uncertainties in the relationships between the nodes) is quantified by the conditional probability distributions. These values are encoded as Conditional Probability Tables (CPT) at each node[7]. The CPT describes the probability that the node will take on its mutually exclusive values (for example, true or false for a binary random variable) given the values of its parents. A Bayesian network thus consists of a qualitative part (the nodes and the links/connections between them representing the structure of the problem being modeled) and a quantitative part (the probabilistic knowledge associated

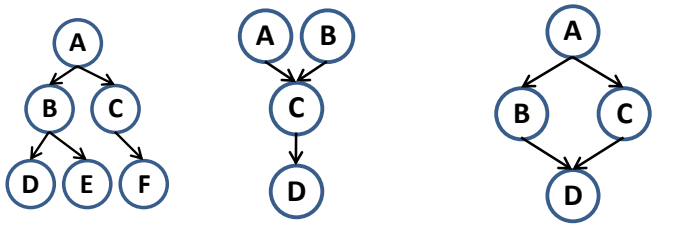
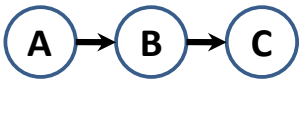


Figure 2-1: Types of Bayesian Network Structures

[7] For discrete variables

with each node in the network). The following section now provides some formal mathematical definitions for a Bayesian network and its components.



	P(A)
$A=a_1$	0.3
$A=a_2$	0.7

P(B|A)		
	$A=a_1$	$A=a_2$
$B=b_1$	0.9	0.4
$B=b_2$	0.1	0.6

P(C|B)		
	$B=b_1$	$B=b_2$
$C=c_1$	0.2	0.75
$C=c_2$	0.8	0.25

Figure 2-2: Example Bayesian Network

Definition 2.6: A Bayesian network B for a set of random variables of interest $X = \{X_1, X_2, \ldots X_n\}$ is a pair (G, θ) where G is a directed acyclic graph over all the variables X and θ is the set of Conditional Probability Tables (CPTs), one for each variable in X, representing the parameterization of the nodes [Darwiche, 2009].

[Pearl, 1988] defines such a set of variables X and a joint probability distribution defined over all the variables as a '*probabilistic model*'. The individual entries in the CPT for a node X_i can be represented as $\theta_{X_i|Pa(X_i)}$ where $Pa(X_i)$ represents the set of parent nodes (direct predecessors) of the node X_i [8]. $\theta_{X_i|Pa(X_i)}$ is referred to as a network parameter. Each $\theta_{X_i|Pa(X_i)}$ corresponds to a particular value assigned to X_i and each of its parents (this is referred to as an instantiation). Essentially,

$$\theta_{X_i|Pa(X_i)} = P(X_i|Pa(X_i)) = P(X_i = x_i|Pa(X_i) = pa_i)$$

$$\sum_x \theta_{X_i|Pa(X_i)} = 1 \qquad \text{Eq.(2-6)}$$

i.e. it represents the conditional probability of X_i, conditioned on all the variables in the set $Pa(X_i)$. The values $P(X_i|Pa(X_i))$ are referred to as local probability distributions [Heckerman, 1995]. Figure 2-2 shows an example of a simple Bayesian network consisting of three nodes and their associated CPTs. The joint probability distribution in this case can be simply represented as $P(a, b, c) = P(c|b).\ P(b|a).\ P(a)$

[8] For a node with n states and $i = 1, 2, 3 \ldots k$ parents, and if S_i is the number of states for the i^{th} parent, the size of the CPT is n rows and $\prod_{i=1}^{k} S_i$ columns

2.4.2. Properties of Bayesian Networks

In general, modeling any domain using a Bayesian network necessitates the assumption of the Markov property which is defined as follows [Neapolitan, 2003]:

Definition 2.7: Given a joint probability distribution P over the random variables in a set X and a directed acyclic graph $G = (X, E)$ where the elements of X are the nodes and the elements of E are the edges (ordered pairs of distinct elements of X), (G, P) is said to satisfy the *Markov condition* if for each variable $X_i \in X$, $\{X_i\}$ is conditionally independent of the set of all its non-descendents given the set of all its parents.

If $ND(X_i)$ represents the set of all non-descendents of X_i, the Markov property may be represented as $I(\{X_i\}, ND(X_i)| Pa(X_i))$. Based on this property, an alternative definition for a Bayesian network may be provided as follows [Neapolitan, 2003]:
Given a joint probability distribution P over the random variables in a set X and a DAG, $G = (X, E)$, (G, P) is said to be a Bayesian network if it satisfies the Markov condition.

The conditional independence assumption allows for a compact representation of the joint probability distribution over X as a product of local distributions which in turn helps in the development of efficient probability propagation algorithms. Mathematically, the joint probability distribution of a set of variables $X= \{X_1, X_2,\ldots X_n\}$ encoded by the Bayesian network can now be represented as:

$$P(X) = \prod_{i=1}^{n} P(X_i|Pa(X_i)) \qquad \text{Eq.(2-7)}$$

Eq.(2-7) is also known as the chain rule of probability [Korb and Nicholson, 2004]. Bayesian networks which have the Markov property are referred to as Independence maps or *I*-maps, wherein all the conditional dependencies are modeled explicitly by the arcs in the network and every independence that actually exists in the domain of interest is suggested by the absence of an arc between the corresponding nodes (Figure 2-3(a))[Korb and Nicholson, 2004]. On the other hand, if every arc in a network indicates a direct dependence between the connected nodes in the domain of interest, then such a Bayesian network is referred to as a Dependence map or a *D*-map [Pearl, 1988].

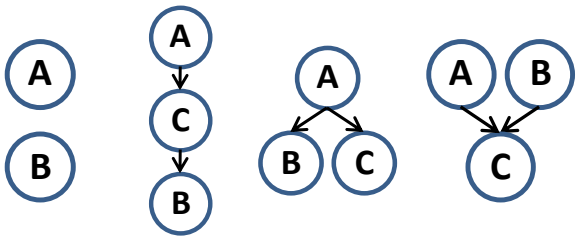


Figure 2-3: Types of Conditional Independence

A network that is both an *I*-map and a *D*-map is said to be a perfect map. The idea of conditional independence represents fundamental qualitative information about the variables in the domain of interest that may be exploited during the propagation of evidence/uncertainties. In particular, there are three types of conditional independencies that are important in determining how the influence of different variables travels across the network and can be interpreted from the topology of the Bayesian network as described in [Pearl, 1988], [Korb and Nicholson, 2004].

i. Serial / Causal Chains

Consider the Bayesian causal network in Figure 2-3(b), where *A* causes *C* which in turn causes *B*. The structure represents the conditional independence relation

$$P(B|A,C) = P(B|C) \equiv A \perp B|C \qquad \text{Eq.(2-8)}$$

Any knowledge of *A* will determine the state of *C* which in turn will influence the value of *B*. Conversely, any evidence about the state of *B* influences the value of *C* which in turn updates the belief in its cause *A*. If, however, any state of *C* is set as evidence, then knowing this value is sufficient to determine the probabilities of various states of *B* (using its CPT that contains conditional probability information representing the link *C*->*B*). Any available evidence about the state of *A* is neither required nor influences the value of *B* any longer. Conversely, by instantiating *C*, any knowledge of *B* will not influence the value of *A*. Thus, in this serial/causal chain *A* and *B* are conditionally independent and separated by *C*.

ii. Common Causes / Divergent Connections

Consider a Bayesian causal network where *A* causes both *B* and *C* (Figure 2-3(c)). This structure results in the same conditional independence relation as before

$$P(C|A, B) = P(C|A) \equiv C \perp B|A \qquad \text{Eq.(2-9)}$$

Any knowledge of *B* influences the value of its cause *A*, which in turn, determines the belief in the value of its other effect *C*. However, if information regarding *A* is known and is used for instantiation, then any additional information regarding the value of *B* does not contribute in any form to the value of *C*. The value of *C* in such a situation is solely determined by using its CPT which contains the conditional probability information representing the link *A*->*C*. Thus, with a common cause type of structure, the nodes *B* and *C* are conditionally independent and separated by *A*.

iii. Common Effects / Convergent Connections

In this case, the effect node *C* has more than one cause, *A* and *B* (Figure 2-3(d)). The conditional independence provided by such a structure is represented as:

$$P(A|B, C) \neq P(A|C) \equiv \neg (A \perp B \mid C) \qquad \text{Eq.(2-10)}$$

The symbol ¬ represents a logical *NOT*. The above equation can be interpreted as follows. The nodes *A* and *B* are independent until the value of *C* is known. However, once the value of *C* is known *A* and *B* become dependent. In other words, with such a structure, any information about the common effect makes its parent nodes which are previously marginally independent, become dependent [Korb and Nicholson, 2004].

The different types of conditional independencies discussed above for pairs of nodes can be extended to a set of nodes using the notion of *d-separation* as follows [Korb and Nicholson, 2004][Russel and Norvig, 1995].

Definition 2.8: An (undirected) path between two sets of nodes *X* and *Y* can be defined as a sequence of nodes between a node in *X* and a node in *Y* such that each adjacent node-pair is connected by a link (without considering the direction of the link) and no node appears more than once in the sequence.

Definition 2.9: If *X*, *Y*, *Z* represent three distinct and disjoint subsets of nodes in a DAG, then *Z* can be said to d-separate *X* from *Y*, if along every path from a node in *X* to a node in *Y* there is a node *W* such that

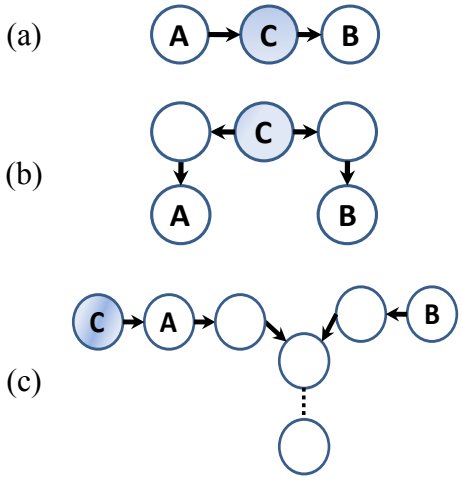


Figure 2-4: d-Separation

i) *W* has converging links (common effect) and neither *W* nor any of its descendants are in the subset *Z*

ii) *W* does not have converging links (common cause or causal chain) and is included in the subset *Z*

In both cases, if the value of *W* is unknown then the path between *X* and *Y* can be said to be *active* whereas any knowledge of *W* is said to *block* the path. Such a set of nodes *Z* is then said to *d-separate* the two sets of nodes *X* and *Y* if every path from a node in *X* to a node in *Y* is blocked given *Z*. In Figure 2-4, the node *C* can be said to d-separate *A* from *B* in each case.

2.4.3. Developing Bayesian Networks

As seen thus far, using a Bayesian network for modeling and decision making in complex problems offers a number of benefits over other techniques, since they not only provide an intuitive graphical representation of the relationships among the variables but also provide a relatively simple yet powerful mathematical structure based on probability theory for quantifying, combining and propagating uncertainties. Explicit representation of the independence and conditional relationships between the variables of interest is especially useful when humans are involved in the decision making loop, as is the case in intelligent EMAs using a performance maps based decision making methodology [Ashok and Tesar, 2007]. Since any good decision made regarding the domain typically needs to take into account all the variables involved, the calculation of joint probabilities is often a major consideration which is greatly simplified by the use of Bayesian networks.

Developing a Bayesian network for a given application involves three main steps:

- Identifying the pertinent variables of interest to be represented as nodes along with their possible states /values that these variables may assume.
- Identifying relationships between the variables and representing them graphically by connecting the appropriate nodes with the links directed in a specific manner
- Quantitatively characterize these relationships i.e. populate the CPTs for all the nodes in the network with the appropriate probability distributions.

2.4.3.1. Network Structure

[Darwiche, 2009] identifies three methods by which Bayesian networks are constructed typically when trying to model and reason about any system. The first method relies on the knowledge of the domain experts to create the structure of the network. The second method synthesizes the network structure from some formal knowledge (especially when the resultant network is used for system analysis). [Darwiche, 2009] refers to these two approaches as the 'knowledge representation approach'. The third approach is to learn the network structure from an available data set using machine learning algorithms (referred to as data-based approach by [Nadkarni and Shenoy, 2004][9]) for example, as described in [Liu, 2008]. Within each approach, the probabilities associated with the different variables represented in the network may be elicited by subjective means from the domain experts or derived from available data.

In the present work, the knowledge representation approach will be used to develop the structure of the Bayesian network. [Nadkarni and Shenoy, 2004] emphasize the fact such an approach is particularly useful when extensive domain knowledge is readily available and eliciting qualitative knowledge from experts is possible to develop comprehensive causal networks. For many engineering systems, different researchers explore a few particular aspects of the system in great detail while not accounting for other facets that may still play an important role in influencing the system behavior.

[9] [Heckerman, 1995], [Acid, et. al, 2004][Neapolitan, 2003][Korb and Nicholson, 2004] provide an extensive treatment of learning algorithms used to learn the network structure as well as probabilities.

However, when taken as a combination, these different fragments of knowledge can be used to develop a truly comprehensive model of the system that may be used for effective operational decision making. Once the qualitative structure is available, the quantitative aspects associated with the structure may be estimated using data (for instance, obtained from extensive testing) to develop very effective Bayesian causal networks. The process of constructing a Bayesian network for a switched reluctance motor and some criteria that may be used to aid the procedure are discussed in detail in Chapters 3 and 4 respectively.

2.4.3.2.Quantifying Relationships

Once the structure of the Bayesian network has been determined, the next step is to quantify the strength of the relationship between the connected pairs of nodes in the form of appropriate conditional probability distributions (for the root nodes this takes the form of prior probability distributions). In other words, to completely specify the network and enable calculation of the joint distribution, it is essential to determine the value of $\theta_{X_i|Pa(X_i)} = P(X_i|Pa(X_i))$ for each node X_i in the network. In order to do this, it is necessary to consider the number of parents of each node and all possible combinations of values of all such parent nodes. Every combination in this set of parents is referred to as an instantiation [Korb and Nicholson, 2004]. Then, for each such instantiation of the parent nodes, the conditional probability for each state of the child node needs to be specified. In case of root nodes, the CPT contains only the values representing the prior probabilities of the variable that it represents. Evidently, as the number of parent nodes increase or if the parent nodes have a large number of states, the size of the CPT increases exponentially.

The parameterization of the different node probabilities is done using the same extensive testing data set that is used in the generation of performance maps. For the present work, data for the Bayesian network has been collected through rigorous testing procedures similar to those described in [Janardhan and Tesar, 2008] and [Yoo and Tesar,

2004][10]. The test set-up described in Appendix B is used for this purpose, by operating it through a complete range of control parameters[11]. There are a number of possible ways to estimate the CPT parameters- using methods like canonical models as described in [Kokkonen, 2005] who employ a generalized Noisy-*OR* model or weights to represent the relative influence of variables as described in [Das, 2004], in order to simplify elicitation from experts or using available data in conjunction with algorithms like Gibbs sampling or Expectation Maximization (EM) [Korb and Nicholson, 2004]. [Murphy, 1999] states that a frequentist method is an oft-used approach to estimate the conditional probability values in the CPT. In the present work too, since all the nodes in the network are assumed to represent discrete variables, the probability distributions are estimated using histograms and a relative frequency approach as follows[12].

Consider a data set consisting of n samples of data, corresponding to the discrete random variable X with k values x_i. Typically, in order to construct a histogram, the range of the variable of interest is divided into a number of intervals, referred to as bins. The states x_i that X can assume are taken to be centered on the bin limits (midpoints). If each x_i, occurs m_i times, then the absolute frequency of x_i is said to be m_i and the relative frequency f_i is calculated as [Marques, 2007]:

$$f_i = \frac{m_i}{n}\ , i = 1,2,3 \ldots . k \text{ and } \sum_{i=1}^{k} m_i = n \qquad \text{Eq.(2-11)}$$

In other words, the relative frequency of each state is considered a proportion of total number of observed values for the variable that lie within that bin. With a large number of samples, the probability of a particular event x_i can be considered as the limit of its relative frequency [Marques, 2007] i.e.

[10] In addition to CPTs, the data can also be stored compactly by fitting structural equations between pairs of cause-effect variables (representing the links in the Bayesian network representation of the system) using Bayesian regression procedures as described in [Ashok and Tesar, 2007] and [Janardhan and Tesar, 2008].

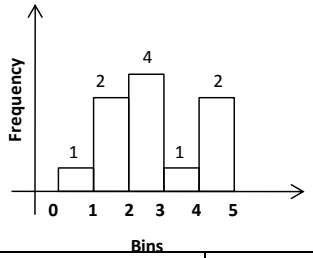
[11] The control parameters for the test set up used in this work include the main supply voltage, the duty cycle and frequency of the pulse width modulation (PWM) signal used to control the motor terminal voltage and the load acting on the motor. The experiments were conducted by varying the supply voltage between 0-20VDC, the PWM duty cycle between 1 and 7 %, and PWM frequency between 2 and 15 kHz.

[12] This is done only to estimate the initial parameters. A method to update these parameters during system operation is presented in [Cohen, et. al, 2001] and will be discussed in Chapter 5.

$$P(x_i) = P(X = x_i) = \lim_{n\to\infty} f_i \qquad \text{Eq.(2-12)}$$

where $P(x_i) \in [0,1]$ and $\sum_{i=1}^{k} P(x_i) = 1$. The values of $P(x_i)$ thus calculated constitute the probability distribution or probability mass function (pmf) of the discrete variable [DeVore, 2000][13] which indicates the probability that a particular state x_i is obtained.

Consider a random variable X such that $0 \leq X \leq 5$. Suppose the variable is discretized into bin sizes of 1 as [0,1],[1,2]….[4,5] . The discrete states of X in this case are X = {0.5, 1.5, 2.5, 3.5 and 4.5}. Suppose ten values of X are obtained over a specific time duration (say, from a sensor used to measure X) as follows X= {0.7, 1.2, 1.6, 2.1, 2.4, 2.6, 2.7, 3.3, 4.5, 4.8}. The histogram for this data is shown in Figure 2-5, where the height of the rectangle is proportional to the absolute frequency of the samples. The relative frequency of the samples is taken as indicative of the probability value of the variable state. Note that the histogram essentially represents the discrete approximation of the variable as mentioned earlier.



	el. Freq.	Prob
P(X=0.5) ⇒P(0≤X≤1)	1/10	0.1
P(X=1.5) ⇒P(1≤X≤2)	2/10	0.2
P(X=2.5) ⇒P(2≤X≤3)	4/10	0.4
P(X=3.5) ⇒P(3≤X≤4)	1/10	0.1
P(X=4.5) ⇒P(4≤X≤5)	2/10	0.2

Figure 2-5: Determining CPTs

2.5. BAYESIAN INFERENCE

Once the Bayesian network has been constructed and all its CPTs have been populated, the model is ready for use in determining different probabilities of interest given the observation of some specific variables in the network. Since a Bayesian network represents a complete joint probabilistic model for all the variables, it can be used to determine updated knowledge of the states of specific variables (posterior distributions) or subsets of variables, by conditioning on the observed values. The process

[13] As the number of samples increases, the empirical histogram (limited samples) obtained in this fashion converges to the actual probability histogram (infinite samples) (law of large numbers).

of determining the posterior probability distributions of the different variables represented by the different nodes in a Bayesian network, given the knowledge of a subset of those nodes, is referred to as 'updating beliefs' or 'probabilistic inferencing' or 'probabilistic propagation' [Korb and Nicholson, 2004].

In essence, the process provides an idea of how probable the different values/ states of a specific variable/node are, using the knowledge already encoded in the CPTs in the network and combining it with newly available knowledge in the form of the observed nodes. The nodes whose values are known or observed are referred to as *evidence* nodes. The subset formed by one or more nodes (other than the evidence nodes) whose posterior distribution values are calculated are referred to as *query* nodes. Depending on which nodes are set as evidence and which nodes are the query nodes, the process of reasoning may be cause to effect (predictive in nature), effect to cause (diagnostic) or combined [Korb and Nicholson, 2004].

2.5.1. Belief Updating in Causal Chains

Consider a simple two node network of the form $A->B$. If there is evidence available about the parent node as $A=a$, then the posterior probability $P(B=b)$ can be determined directly from the CPT of B as $P(B=b|A=a)$. If $B=b$ is the state set as evidence, then the posterior probability $P(A=a)$ can be easily determined using Bayes' theorem (Eq.(2-3)). With a three node network of the form $A->B->C$, the calculations are slightly more complex. In this case, if there is evidence in the form $A=a$, then, $P(C=c|A=a)$ is determined using the implied conditional dependencies and the chain rule of probability (Eq.(2-7)) as:

$$P(C = c|A = a) = \sum_{b} P(C|B)P(B|A = a) \qquad \text{Eq.(2-13)}$$

Similarly the value of $P(A=a|\ C=c)$ can again be calculated using the chain rule of probability and Bayes' theorem. But, for more complex Bayesian network structures, the process of belief updating can be quite complex.

2.5.2. Belief Updating in Polytree and Multiply Connected Networks

Various algorithms have been developed by researchers for belief updating in singly connected networks or polytrees as well as multiply connected networks. The algorithms are broadly classified into two categories as exact (e.g. Pearl's message passing algorithm [Pearl, 1988], the Lauritzen-Spiegelhalter clique tree propagation algorithm [Lauritzen and Spiegelhalter, 1988], the symbolic probabilistic inference (SPI) algorithm [Li and D'Ambrosio, 1994], etc.) and approximate inferencing algorithms (e.g. Logic Sampling [Henrion, 1988], Likelihood Weighting [Fung and Chang, 1989], [Shachter and Peot, 1989], Loopy Belief Propagation [Murphy, et. al. 1999], etc.). [Guo and Hsu, 2002] provide a comprehensive review and comparison of different such types of algorithms. Some algorithms are appropriate only for sparse networks, while some can be applied only to polytree structures. The choice of an appropriate algorithm to use is determined by factors like the structure of the Bayesian network, computational requirements, etc.[14]. The following section describes three popular algorithms in detail.

2.5.2.1.Exact Inferencing

2.5.2.1.1. Pearl's Belief Propagation Algorithm

[Pearl, 1988] developed a message-passing exact inferencing algorithm to propagate beliefs in polytree networks[15]. The algorithm makes use of the local independencies between the nodes as embodied by the structure of the Bayesian network and Bayes' theorem to determine the posterior probability distributions of the query nodes in the network.

Consider a Bayesian network comprised of a set of nodes $X = \{X_1, X_2, .. X_n\}$ where n is the total number of nodes in the network. Suppose the values $E=e$ are specified for a set of evidence nodes $E= \{E_1, E_2, \ldots E_m\}$ i.e. $\{E_1= e_1, E_2=e_2 \ldots\ldots E_m=e_m\}$, where $E \subseteq X$ and m is the number of nodes set as evidence to the network. The objective

[14] The problem of exact as well as approximate inferencing in Bayesian networks were proven to be NP-hard by [Cooper, 1990] and [Dagum and Luby, 1993] respectively; indicating the computational challenges involved even for moderately sized networks where the nodes have a limited number of states.

[15] The core algorithm is for singly connected networks but has been extended to apply to multiply connected networks as well using methods like conditioning /clustering, etc.

is thus to determine the posterior distribution for one or more query nodes X_i. In order to do this, two types of parameters known as the $\lambda(X_i)$ and $\pi(X_i)$ values are used in the conjunction with the CPT of a node X_i to update its belief. The values of these parameters are calculated based on two types of messages- λ and π that X_i receives. The λ messages are sent from a child node to all its parent nodes (i.e. against the direction of the links in the network) and are represented using the notation $\lambda_{Child}(Parent)$ i.e. $\lambda_{Sender}(Reciever)$ [Korb and Nicholson, 2004] The π messages are sent from a parent node to all its child nodes (i.e. in the direction of the links in the network)and are again represented as $\pi_{Child}(Parent)$ i.e. $\pi_{Receiver}(Sender)$ [Korb and Nicholson, 2004]. Note that every node X_i in the network sends and receives these messages. The notations used in the algorithm are shown in Figure 2-6. Let $W_{1\ldots p}$ represent the set of parent nodes for a given node X_i. Let $Y_{1\ldots q}$ represent the set of child nodes for X_i. Let Z represent the set of other parent nodes for a given child node Y of X_i. [Korb and Nicholson, 2004] described the three major components of the algorithm as: belief updating, bottom-up propagation and top-down propagation, which are discussed below.

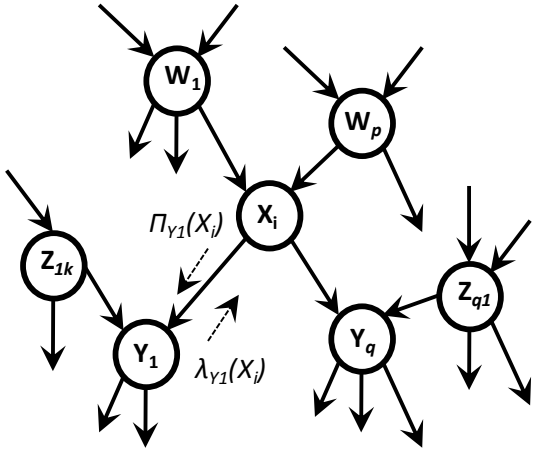


Figure 2-6 : Pearl's Belief Propagation Algorithm

In bottom-up propagation, the λ messages sent from a node to all its parents are computed. For every child Y of the node X_i, the λ message sent from Y to X_i, for every value x of X_i, is calculated as [Neapolitan, 2003]

$$\lambda_Y(x) = \sum_y \lambda(y) \left[\sum_{z_1,\, z_2 \ldots . z_k} \left(P(y|z_1,\, z_2 \ldots \ldots z_k) \prod\nolimits_{i=1}^{k} \pi_Y\,(z_i) \right) \right] \qquad \text{Eq.(2-14)}$$

where y are all the values of Y, $Z_1 \ldots Z_k$ are the parents of Y other than X, with values z.

The λ values are computed as follows [Neapolitan, 2003]:

- If $X_i \in E$ i.e. it is an evidence node and if the state $\hat{x}$ is the state set as evidence,

$$\lambda(\hat{x}) = \begin{cases} 1 \\ 0 \ \ for\ x \neq \hat{x} \end{cases} \quad \text{Eq.(2-15)}$$

- If $X_i \notin E$ (i.e. not an evidence node) but is a leaf node, then for all values x,

$$\lambda(x) = 1 \quad \text{Eq.(2-16)}$$

- If $X_i \notin E$ (i.e. not an evidence node) and it is also not a leaf node, then for all values x

$$\lambda(x) = \prod_{Y_q \in Child_{X_i}} \lambda_{Y_q}(x) \quad \text{Eq.(2-17)}$$

where $Child_{X_i}$ represents the set of all the child nodes of X_i.

In top-down propagation, the π messages sent from a parent node to all its children are calculated. If W_p is any parent of the node X_i, then for all the values x of X_i, then the π messages are given by[Neapolitan, 2003]

$$\pi_X(w) = \pi(w) \prod_{V \in Child_{W_p} - \{X_i\}} \lambda_V(w) \quad \text{Eq.(2-18)}$$

where $Child_{W_p} - \{X_i\}$ represents the set of children of a node W_p other than X_i.

The π values are calculated as follows [Neapolitan, 2003]:

- If $X_i \in E$ (i.e. an evidence node) and if the state $\hat{x}$ is the state set as evidence, then

$$\pi(\hat{x}) = \begin{cases} 1 \\ 0 \ \ for\ x \neq \hat{x} \end{cases} \quad \text{Eq.(2-19)}$$

- If $X_i \notin E$ (i.e. not an evidence node), but it is a root node, then for all values x

$$\pi(x) = P(x) \quad \text{Eq.(2-20)}$$

- If $X_i \notin E$ (i.e. not an evidence node), and it is also not a root node, then for all the parents $W_1, W_2, \ldots W_p$ of X_i (with values $w_1, w_2, \ldots . w_p$) and for values x of X_i.

$$\pi(x) = \sum_{w_1, w_2 \ldots . w_p} \left(P(x|w_1, w_2 \ldots . w_p) \prod_{i=1}^{p} \pi_X(w_i) \right) \quad \text{Eq.(2-21)}$$

Finally, the belief for a node X_i is updated when the messages arrive from its children and parent nodes, indicating a modification in their respective belief values. The updated posterior probability for the different states x of X_i is given by [Neapolitan, 2003]:

$$P(x|e) = \alpha\lambda(x)\pi(x) \qquad \text{Eq.(2-22)}$$

where α is a normalizing constant which ensures that $\sum_x P(X_i|E) = 1$

With all the above computations, the algorithm can now be summarized using the pseudo-code, as described in [Neapolitan, 2003], as follows (this algorithm has also been implemented in C++ as part of the AMOS libraries (Appendix C) for this work):

Objective Given a Bayesian network consisting of a set of nodes $X = \{X_1, X_2, .. X_n\} = \{X_i\}$, where $i = 1, 2, \ldots n$, n is the total number of nodes in the network with a singly connected or a tree type of structure and a set of evidence nodes, determine the posterior probability values for the query nodes

Inputs The specified set of values $E=e$ of a set of evidence nodes $E= \{E_1, E_2, \ldots E_m\}$ i.e. $\{E_1 = e_1, E_2=e_2 \ldots \ldots E_m=e_m\}$, where $E \subseteq X$ and m is the number of nodes set as evidence to the network

Outcome The conditional posterior probability values for all the query nodes in the network i.e. computing $P(X_j \mid E)$ where $X_j \in (X - E)$

Algorithm

1. For every node in the X_i network

   Loop through i for number of nodes X_i

   Loop through number of states j for the X_i^{th} node

   Set all $\lambda(x) = 1$

   Exit Loop j

   Loop through number of parents p for the X_i^{th} node

   Loop through number of states l for the W_p^{th} node

   Set all $\lambda_X(w) = 1$

Exit Loop l

Exit Loop p

Loop through number of children q for the X_i^{th} node

Loop through number of states l for the Y_q^{th} node

Set all $\pi_Y(x)$ =1

Exit Loop l

Exit Loop q

Exit Loop i

2. For the root nodes

Loop through number of root nodes p

Loop through number of states k for the p^{th} node,

Set all $\pi(w) = P(w)$, the prior probability

Exit Loop j

Loop through number of children i for the p^{th} root node

Send_ π_Message from W_p to X_i

Exit Loop i

Exit Loop p

3. Update the Network

Given a set of evidence nodes $E= \{E_1, E_2, \ldots E_m\}$ i.e. $\{E_1= e_1, E_2=e_2 \ldots\ldots E_m=e_m\}$

Loop through i for number of nodes X_i

If $X_i \in E$ and if the state $\hat{x}$ is the state set as evidence

Set $\lambda(\hat{x})$ =1, $\pi(\hat{x})$ =1, $P(\hat{x}|e)$ =1

For all other states x of X_i where $x \neq \hat{x}$

Set $\lambda(x)$ =0, $\pi(x)$ =0, $P(x|e)$ =0

Exit Loop i

4. Definitions of the Send_π_Message and Send_ λ_Message functions

Send_π_Message from a child Y_q to X_i

For all k state values x of X_i,

Calculate $\lambda_Y(x)$ using Eq.(2-14)

Calculate $\lambda\ (x)$ using Eq.(2-17)

Compute $P(x|e)$ using Eq.(2-22)

Exit Loop k

Normalize $P(x|e)$

For each parent W_p of X_i such that $W_p \notin E$, Send_ λ_Message from X_i to W_p

For each child YY_q of X_i such that $YY_q \not\equiv Y_q$, Send_ λ_Message from X_i to YY_q

Send_λ_Message from a parent W_p to X_i

For all k states w of Wp

Calculate $\pi_X(w)$ using Eq.(2-18)

If $X_i \notin E$, calculate $\pi(x)$ for all states x of X_i using Eq.(2-21)

Compute $P(x|e)$ using Eq.(2-22)

Exit Loop k

Normalize $P(x|e)$

For each child Y_q of X_i, Send_ λ_Message from X_i to Y_q

If at least one state of X_i or all its descendents is instantiated (i.e. $\lambda(x) \neq 1$ for all states x of X_i), then

For each parent WW_p of X_i such that $WW_p \notin E$ and $WW_p \not\equiv W_p$

Send_ λ_Message from X_i to WW_p

2.5.2.2. Approximate Inferencing

Even though algorithms like loopy belief propagation, clustering, etc. are efficient even for multiply connected networks, the computational complexity increases tremendously for networks with a large number of nodes or for networks that are densely connected [Korb and Nicholson, 2004]. This led to the development of approximate inferencing algorithms. The two primary approaches to do this are by using stochastic simulation and by using deterministic search [Neapolitan, 2003][16]. In stochastic

[16] [Guo and Hsu, 2002] provide a detailed survey of other approaches used for approximate inferencing

simulation, inferencing is accomplished by sampling data based on the probability distributions of the different nodes in the network (in other words, a large number of different cases are generated to simulate the process of sampling from the network distribution[17]). The generated samples are then used to determine the posterior probability distributions of the nodes of interest. As more and more cases are generated, the estimated posterior probability distributions of the different nodes converges to the exact distributions based on the law of large numbers from classical statistics [Korb and Nicholson, 2004]. The following section discusses two such algorithms that are commonly used, the Logic Sampling algorithm and the Likelihood Weighting algorithm.

2.5.2.2.1.Logic Sampling

The Logic Sampling algorithm was originally developed by [Henrion, 1988]. The algorithm begins with an ancestral ordering of nodes[18]. The algorithm traverses each node in the network, beginning from the root nodes down to the leaf nodes based on this ordering. At each step, a value (state) is randomly generated for every node at the existing ordering level based on their conditional probability distributions (for root nodes, this corresponds to the prior distribution and for any other node in the network, this corresponds to the appropriate CPT value based on the sample values that are generated for all the parents of that node).

If during the process, the value generated for an evidence node does not coincide with the value that has already been specified for it, the algorithm restarts by generating fresh random values for all the nodes starting again with the root nodes and working its way through the network. The process continues until a value has been successfully generated for every node, which happens only if the correct values are generated for all the evidence nodes. When all the nodes have been visited in this manner, then it can be

[17] For instance, using a random number generator that produces values between 0 and 1. The random number generator used in the AMOS implementation of approximate inferencing algorithms (Logic Sampling and Likelihood Weighting) is the Mersenne Twister fast implementation developed by [Matsumoto, M. and Nishimura,T.,1998] and ported to C++ as described in [Bedaux, J., 2008]

[18] An arrangement of nodes in which, if node *B* is a descendant of node *A*, then *B* occurs later in the ordering than *A* as <*A*, *B*>

said that a successful instantiation has been obtained for one trial or case [Korb and Nicholson, 2004]. The above procedure is repeated for a specified number of trials (usually specified based on some user-defined convergence criteria for a given application). The final posterior probability distributions for every node in the network are then calculated based on the number of times the different states of each of those nodes are successfully generated as samples within the specified number of trials.

The aspect of rejecting cases for which the values generated for evidence nodes are not favorable, is often cited as the major drawback of the Logic Sampling algorithm. This is because, if the evidence for any node in the set of evidence nodes E is unlikely (i.e. $P(E=e)$ is very low), then each time a random value for that node is generated in a particular trial and it does not correspond to the state e, then all such cases will have to be rejected with a probability of 1-$P(e)$ [Neapolitan, 2003] . In other words, these cases (i.e. samples that are inconsistent with the observed evidence) will be discarded and will not contribute towards the convergence of the probability distributions for the query nodes under consideration. This results in a much slower convergence of the estimated probability values to the true distributions (since a large number of samples need to be generated to result in sufficient number of usable values).

The Logic Sampling algorithm implemented as part of the AMOS C++ library is based on the pseudo-code provided in [Neapolitan, 2003] and is described in the following section using the same.

Objective Given a Bayesian network consisting of a set of nodes $X = \{X_1, X_2, .. X_n\}$ where n is the total number of nodes in the network and a set of evidence nodes, determine the posterior probability values for the query nodes

Inputs

- The specified values $E=e$ of a set of evidence nodes $E= \{E_1, E_2, ... E_m\}$ i.e. $\{E_1= e_1, E_2=e_2 ..... E_m=e_m\}$, where $E \subseteq X$ and m is the number of nodes set as evidence to the network
- Number of trials t

Outcome The conditional posterior probability values for all the query nodes in the network i.e. computing $P(X_j \mid E)$ where $X_j \in (X - E)$

Algorithm

1. Order the n nodes in the network in an ancestral ordering
2. Loop through j for number of nodes X_j where $X_j \in (X - E)$

   For each node X_j, with k states (i.e. $x_{j1}, x_{j2}, \ldots x_{jk}$) in the network

   a. Create a variable, say, $Count_{X_{jk}}$ to track the number of times each state k of X_j is generated during the sampling process

   b. Initialize $Count_{X_{jk}}$ for each $x_{jk} = 0$

   Exit Loop j
3. Loop through number of trials t (user-specified)

   For each new trial

   a. Loop through the number of nodes n starting from j=1 and while j <= n

      i. Generate a value ($\tilde{x}_j$) for the node X_j knowing the values generated for its parents $Pa(X_j)$ in the previous step i.e. $\widetilde{pa}_j$ and using the conditional probabilities from the CPT i.e. $P(x_j \mid \widetilde{pa}_j)$

      (for root nodes generate random values using their priors)

      ii. If $X_j \in E$ with x_{jk} as the instantiated state, if $\tilde{x}_j = x_{jk}$,

      Go to the next node in the ancestrally ordered list (increment j)

      If not, reset j =1

   Exit Loop j

   b. If the values for all the nodes have been generated successfully,

   Loop through j for number of nodes X_j where $X_j \in (X - E)$

   Loop through number of states k for the X_j^{th} node

   If the sample generated $\tilde{x}_j$ is the same as the value of the k^{th} state x_{jk} for the X_j^{th} node, increase the corresponding count variable $Count_{X_{jk}}$ by 1

Exit Loop *k*

Exit Loop *j*

Exit Loop *t*

4. Loop through *j* for number of nodes X_j where $X_j \in (X - E)$

   Loop through number of states *k* for the X_j^{th} node

   For each state *k* of the node X_j, calculate the estimated posterior probability value $\tilde{P}(x_{jk}|E)$ as: $\tilde{P}(x_{jk}|E) = \frac{Count_{X_{jk}}}{t}$

   Exit Loop *k*

Exit Loop *j*

The algorithm described above can be applied to any number of query nodes. In order to assess the quality of convergence [Korb and Nicholson, 2004] suggest the use of appropriate distance measures to quantify the difference between the true distribution *P* for a random variable represented by a node, and its estimated distribution $\tilde{P}$. One such measure, the Kullback-Leibler (KL) divergence between *P* and $\tilde{P}$ (assuming discrete distributions) is given by:

$$KL\ Divergence = \sum_{i=1}^{k} P(i) log \frac{P(i)}{\widetilde{P(i)}} \qquad \text{Eq.(2-23)}$$

2.5.2.2.2.Likelihood weighting

In order to mitigate the shortcomings of the Logic Sampling algorithm when dealing with unlikely evidence values, and to make use of every sample that is generated, the Likelihood Weighting algorithm was proposed [Fung and Chang, 1989], [Shachter and Peot, 1989]. In this case, instead of incrementing the $Count_{X_{jk}}$ variable by 1 only when a favorable sample is generated (i.e. one which is consistent with the available evidence values), the variable is incremented by an amount which represents the fractional likelihood of occurrence of the evidence values (i.e. how likely are the sample values generated for the evidence nodes, based on their CPTs [Korb and Nicholson,

2004]). This incremental amount is referred to as *score* by [Neapolitan, 2003] and is defined as follows. Consider a Bayesian network consisting of a set of nodes $X = \{X_1, X_2,... X_n\}$ where n is the total number of nodes in the network. Let the values be specified for a subset of m evidence nodes $E = \{E_1, E_2,... E_m\}$ where $E \subseteq X$ as $\{E_1 = e_1, E_2 = e_2 ... E_m = e_m\}$. Let X_j be the set of nodes such that $X_j \in (X - E)$. The value of *score* is then defined as [Neapolitan, 2003]:

$$score(\tilde{x}_j) = \prod_{X_i \in E} P(\tilde{x_i} | \tilde{pa_i}) \qquad \text{Eq.(2-24)}$$

Where $\tilde{x_i}$ and $\tilde{pa_i}$ are the values generated during the sampling process for the nodes X_i and its parents PA_i which belong to the set of evidence nodes E, and $\tilde{x}_j$ represents the set of values generated for the variables in X_j. The Likelihood Weighting Algorithm implemented as part of the AMOS libraries is based on the pseudo-code provided in [Neapolitan, 2003] and is described below using the same.

Objective Given a Bayesian network consisting of a set of nodes $X = \{X_1, X_2, .. X_n\} = \{X_i\}$, where $i = 1, 2, \ldots n$, n is the total number of nodes in the network and a set of evidence nodes, determine the posterior probability values for the query nodes

Inputs

- The specified set of values $E=e$ of a set of evidence nodes $E = \{E_1, E_2,... E_m\}$ i.e. $\{E_1 = e_1, E_2 = e_2 \ldots\ldots E_m = e_m\}$, where $E \subseteq X$ and m is the number of nodes set as evidence to the network
- Number of trials t

Outcome The conditional posterior probability values for all the query nodes in the network i.e. computing $P(X_j \mid E)$ where $X_j \in (X - E)$

Algorithm

1. Order the n nodes in the network in an ancestral ordering
2. Loop through j for number of nodes X_j where $X_j \in (X - E)$

   For each node X_j, with k states (i.e. $x_{j1}, x_{j2}, \ldots x_{jk}$) in the network

Initialize the estimated posterior probabilities for each state k of X_j as

$\hat{P}(x_{jk} \mid e) = 0$

End Loop j

3. Loop through m for number of nodes X_m where $X_m \in E$

   Clamp the estimated value of X_m i.e. $\tilde{x}_m$ to the value of E_m in e

   End Loop m

4. Loop through number of trials t (user-specified)

   For each new trial

   a. Loop through the number of nodes n starting from j=1 and while j <= n

      Generate a value ($\tilde{x}_j$) for the node X_j knowing the values generated for its parents $Pa(X_j)$ in the previous step i.e. $\widetilde{pa}_j$ and using the conditional probabilities from the CPT i.e. $P(x_j \mid \widetilde{pa}_j)$

      (for root nodes generate random values using their priors)

      Exit Loop j

   b. Calculate the score based on the clamped evidence values

$$score(\tilde{x}_j) = \prod_{X_i \in E} P(\tilde{x}_i | \widetilde{pa}_i)$$

   c. Loop through j for number of nodes X_j where $X_j \in (X - E)$

      Loop through number of states k for the X_j^{th} node

      If the sample generated $\tilde{x}_j$ is the same as the value of the k^{th} state x_{jk} for the X_j^{th} node, increase the estimated posterior probability by the score as follows:

$\hat{P}(x_{jk} \mid e) = \hat{P}(x_{jk} \mid e) + score$

      Exit Loop k

   Exit Loop j

Exit Loop t

5. Loop through j for number of nodes X_j where $X_j \in (X - E)$

   Loop through number of states k for the X_j^{th} node

Normalize $\hat{P}(x_{jk} \mid e)$

Exit Loop k

Exit Loop j

2.6. Chapter Summary

This chapter has identified the use of Bayesian networks as the method of choice in modeling intelligent EMAs. The key topics discussed in this chapter and the relevant sections are shown in Table 2-2.

Bayesian networks allow the representation of intricate relationships (causal/non-causal) between the different variables of interest in the domain in an explicit manner and also provide a means to account for the uncertainty in each of those variables using probability theory (in the form of conditional probability tables for discrete variables). This not only facilitates the incorporation of the knowledge of domain experts in developing the model for a given system but also allows for the refinement of this knowledge (for instance, via Bayes' theorem) when new data becomes available [Friedman, 1997].

The graphical nature of the technique also provides an intuitive understanding of the resulting model (especially causal influences) and provides a greater insight into the observed behavior of the system when compared to many other approaches where such knowledge is not very easily accessible. This has the advantage of the model being more readily adopted and easy to use when humans are involved in the decision making process, as is the case in intelligent EMAs. The network also has the advantage of being modified with relative ease with the addition of new nodes and links (when the associated data is also available) to account for additional variables of interest. Depending on the resultant structure, the network can be evaluated quickly based on available information (evidence) to make both forward and inverse (predictive and diagnostic) inferences about the other variables in the network (queries) using different belief updating algorithms as described in Section 2.5. The next chapter will now discuss the development of a Bayesian network for the test set-up described in Appendix B.

Table 2-2 : Chapter Summary

Topic	Section	Description
Modeling of Intelligent EMAs	Section 2.1	Requirements of modeling technique • Ability to accurately and compactly represent systems without requiring many simplifying assumptions (like linearization) • Incorporate existing domain knowledge (available data) • Intuitive to understand • Support forward and inverse decision-making • Allow incorporation and propagation of uncertainty in different variables pertinent to the system/application • Fast evaluation and modification/reconstruction • Adaptability to rapidly changing environments
Bayesian Networks	Sections 2.3, 2.4, and 2.5	• Graphical modeling using Directed Acyclic Graphs (DAGs) • Based on well-developed concepts of probability theory • Compact representation of the joint probability distribution of a group of random variables (discrete/continuous) • Explicit representation of probabilistic dependencies (nodes connected by directed links and characterized by conditional probability tables or CPTs) • Updated knowledge of values of different variables in the network based on a subset of observed/evidence values obtained via belief updating • Use of inferencing algorithms to update beliefs — Exact Inferencing e.g. Pearl's Belief Propagation Algorithm — Approximate Inferencing e.g. Logic Sampling, Likelihood Weighting

Chapter 3. Bayesian Network Model of Switched Reluctance Motor

3.1. INTRODUCTION

As seen in Chapter 2, a Bayesian network can provide a unified framework for graphically modeling a system consisting of numerous parameters of interest with relative ease as well as for decision making (for operation, condition monitoring, maximum performance, etc.) in the presence of uncertain knowledge. In this chapter, the development of a Bayesian network to represent a system consisting of a switched reluctance motor, controller, loading brake and instrumented with multiple sensors is explored. The complete model is first presented in Section 3.2 and different segments of the model are explained in greater detail in the different subsections of Section 3.3.

3.2. BACKGROUND

A Switched Reluctance Motor (SRM) is a singly-excited motor with multiple phases consisting of salient stator and rotor poles. The motor has windings only on the stator while the rotor is built up from a stack of steel laminations with no windings or rare earth magnets. A stator phase consists of two series-connected windings on diametrically opposite poles. The different phases in a SRM may be considered as working independently with low mutual coupling between the phases. This simplification is often used to reduce the complexity of analysis in many aspects. The process of conversion of electromagnetic to mechanical energy can be analyzed in a single phase and a superposition of individual values (for each motor phase) of parameters of interest

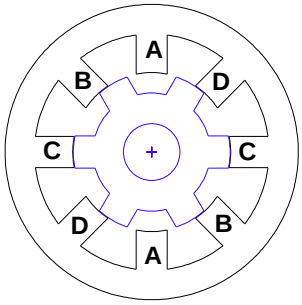


Figure 3-1: Switched Reluctance Motor

like torque, losses, etc. may be done in order to determine their total values[1]. When a pair of stator poles is energized (by current flowing through its phase winding), the pair of rotor poles closest to it is attracted towards it until the rotor comes to a configuration where the two poles are aligned. By switching the successive phases on and off in sequence (a process referred to as commutation), continuous rotation and torque production can be achieved in either direction. The use of SRMs as the choice of prime mover has been explored in many of the intelligent EMA designs developed at the UT-RRG [Tesar, 2009][Ashok and Tesar, 2001] due to the many advantages it offers - brushless operation (low maintenance), good power density and lower losses compared to other motor types for specific sizes, better robustness and reliability, high efficiency over a significant range of operation, capability for high speed operation, potential for fault tolerance (due to electrically separate windings), simpler cooling requirements, etc. (Increased torque ripple, acoustic noise generation, need for complex control electronics are some of the commonly cited disadvantages for SRMs [Ramu, 2001]).

The motor has also been widely studied by various researchers over the years from different perspectives covering both design and operational aspects [Gobbi, et. al., 2007][Omekanda, 2003][Ramu, 2001], [Soares, et. al, 2001][Hatcher and Tesar, 2000], [Miller, T.J.E, 1993], etc. Such research has yielded powerful and comprehensive (but highly nonlinear[2]) mathematical models tailored to different aspects of motor design and control like maximizing torque production/efficiency, reducing acoustic noise generation, etc. However, most of these models are typically meant to investigate motor behavior in simulations. When implemented in real world systems, these individual models are used to address a specific objective and typically do not account for the interaction between the different physical phenomena that affect the motor performance (for instance, as demonstrated in [Ustun, 2009], [Beno, et. al, 2007], [Rahman, et. al, 2001] which explore the use of neural networks, fuzzy logic, etc. to model and control SRMs). Hence, they are

[1] The work of [Kokernak and Torrey, 2000] however demonstrates the benefits that can be achieved by taking the mutual coupling into account.

[2] These nonlinear relations are often stored as look-up tables, for example, rotor position vs. torque/flux

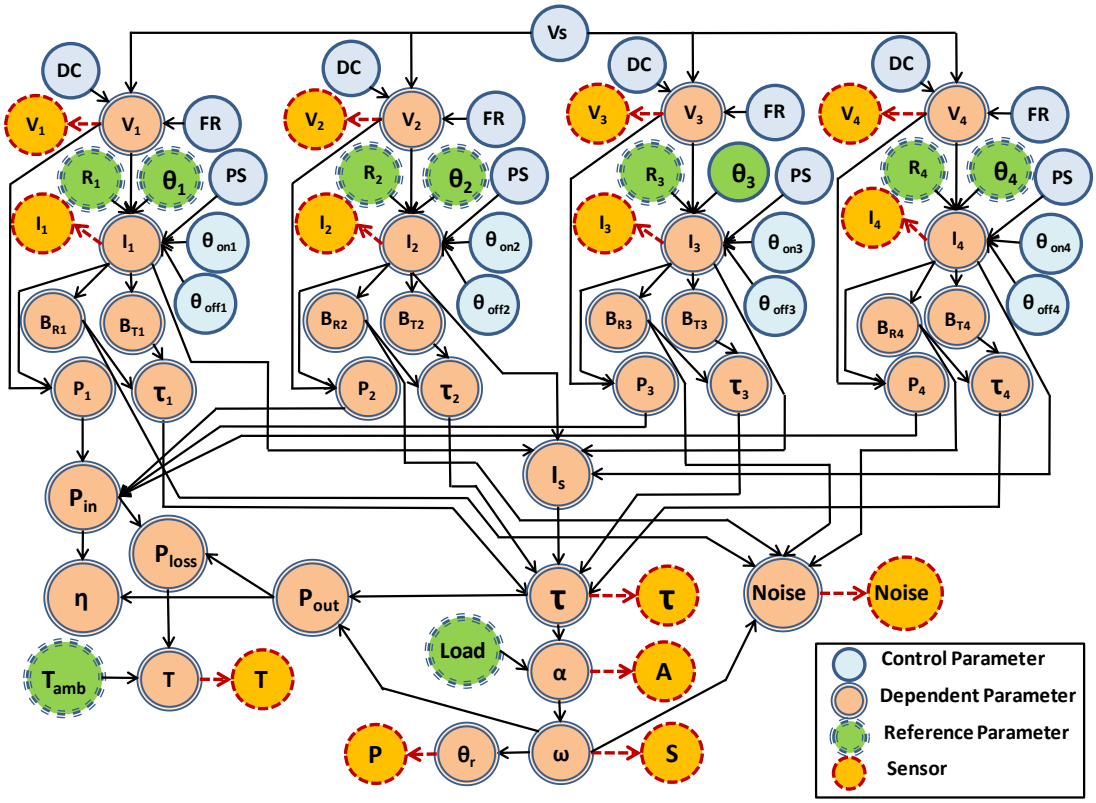
not easily adaptable for direct use in overall system-level decision making or condition monitoring using the performance maps and envelopes based approach. In addition, the uncertainty in various parameters is not taken into account in these approaches.

[Matveev, 2006] demonstrates a multi-disciplinary approach to model a SRM (from a design perspective) that takes into account various aspects like torque production, acoustic noise generation, drive efficiency, thermal considerations, etc. with detailed analyses and mathematical models for each of these aspects as well as the interconnections among them. The work presented in this chapter aims to follow a similar approach, taking into account the different facets mentioned above, to develop a comprehensive SRM model that is useful from the decision making (operational) point of view. Figure 3-2 shows the Bayesian network model of the test set-up used in this research[3] (a detailed description and important specifications for the different components used is provided in Appendix B).

The network structure is derived by combining different segments representing the different models available in literature using the Bayesian network framework. The control parameters (supply voltage, PWM duty cycle and frequency for each phase, phase status and turn-on/turn-off angles), the reference parameters (phase resistances, relative phase angles, brake load and ambient temperature), and the dependent parameters (phase currents, radial and tangential flux densities, torque, speed, acceleration, noise, losses, etc.) are also shown with their respective notations. Note that the network is causal and represents how the variables influence each other. This helps make the network intuitive to the end-user faced with having to make system-level decisions on how to manage the motor performance based on maps and decision surfaces. Correlating the different variables in this manner helps to retain and propagate the uncertainty information associated with each parameter. In addition, it also forms a basis to provide functional redundancy for all the variables involved. This is particularly valuable in providing

[3] The sensor nodes and the links (shown in dotted lines) are not part of the actual network structure. They are shown only to illustrate the variables which can be directly measured or derived using information from the sensors available in the test set-up

alternative sources of information as well as in validating sensor readings. The following section describes the individual network segments in greater detail.



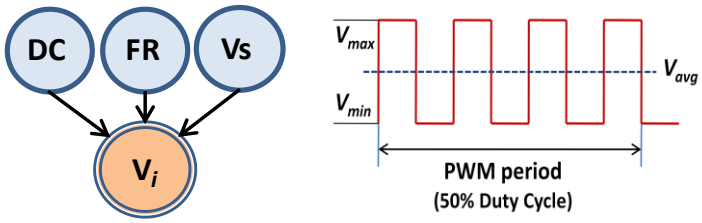
i = 1, 2, 3, 4 = Number of motor phases
V_s = Supply Voltage (Line)
DC = PWM Duty Cycle
FR = PWM Frequency
V_i = Phase Voltage
R_i = Phase Resistance
Θ_{ph} = Rotor Angle Relative to Stator
PS = Phase Status (On/Off)
Θ_{off} = Phase Turn-off Angle
Θ_{on} = Phase Turn-on Angle
I = Phase current
I_s = Supply Current (Line)
B_R = Magnetic flux density, radial
B_T = Magnetic flux density, tangential

τ_i = Torque produced by a single phase
τ = Total Electromagnetic Motor Torque
α = Angular Acceleration of Rotor
ω = Angular Velocity of Rotor
θ_r = Rotor Position
$Load$ = External Load Torque
$Noise$ = Acoustic Noise
P_{in} = Input Electrical Power
P_{out} = Output Mechanical Power
P_{loss} = Total Power Losses
η = Motor Efficiency
T_{amb} = Ambient Temperature
P_{loss} = Total Power Loss
T = Average Motor Temperature

Figure 3-2: Bayesian Network Model for a Four Phase Switched Reluctance Motor

3.3. MODEL DESCRIPTION

3.3.1. Influence of Control Parameters on Motor Phase Voltage

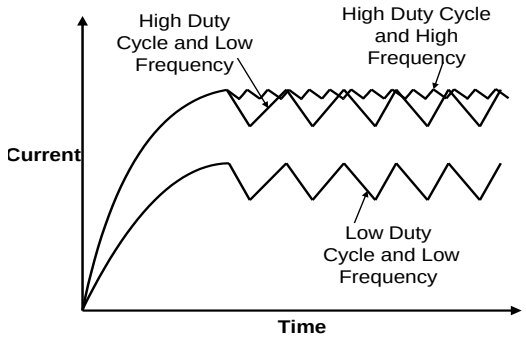


i = 1…n = Number of motor phases,

DC= PWM Duty Cycle,

FR= PWM Frequency, V_s= Supply Voltage

Figure 3-3 : Relation between PWM Duty Cycle, PWM Frequency and Voltage



[Ashok and Tesar, 2007]

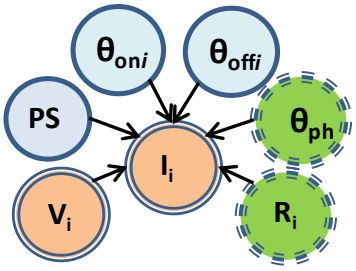
Figure 3-4: Effect of PWM Duty Cycle and Frequency on Phase Current

The 5V TTL Pulse Width Modulated (PWM) input signals from the motor controller are passed to the motor via an opto-isolator bank through the H-bridge amplifier circuit (see Appendix B for details). This input signal is used to pulse the supply voltage (0-60VDC range) which results in the final voltage across the individual motor phases. The current drawn by each phase depends on this voltage. Thus the input PWM signals control the amount of power entering the system. A PWM signal is characterized by its duty cycle and its frequency (Figure 3-3). Duty cycle is the ratio of the "on" time of the pulse to the "off" time of the pulse. The frequency dictates how fast the switching of the signal occurs. If V_{max} and V_{min} are the maximum and minimum voltages across which the signal is pulsed with a duty cycle DC, then, the average voltage is:

$$V_{avg} = DC.Vmax + (1 - DC).Vmin \qquad \text{Eq. (3-1)}$$

A low value for the duty cycle produces a low value of average voltage which in turn results in a lower current flowing through the motor phase and vice –versa. The higher the switching frequency, the smoother is the supplied voltage and smoother is the resulting current profile [Ashok and Tesar, 2007]. Figure 3-4 shows the effect of varying PWM duty cycle and frequency on the phase current profile. Thus, it can be seen that the motor phase voltage (amplitude and quality) depends on the duty cycle and frequency of the PWM input signals and the magnitude of the supply voltage (V_s)[4].

3.3.2. Relation between Phase Voltage, Phase Current and Phase Resistance



i = 1…n = Number of motor phases,

V_i = Phase Voltage, R_i = Phase Resistance,

Θ_{ph} = Rotor Angle Relative to Stator,

PS= Phase Status (On/Off)

Θ_{offi} = Phase Turn-off Angle,

Θ_{oni} = Phase Turn-on Angle

I_i = Phase Current

Figure 3-5 : Effect of Various Control and Reference Parameters on Phase Current

The current drawn by a phase depends on the voltage applied across it, whether it is switched on/off based on its occurrence in the commutation sequence, its resistance, the mechanical rotor angle (relative to the stator) and the turn-on and turn-off angles for the phase as shown in Figure 3-5. The phase voltage is applied across the windings only when the particular phase is switched on or off during the commutation sequence. The equation governing the phase voltage across the stator windings (neglecting the nonlinear effects of saturation and mutual fluxes) relates it to the resistive voltage drop and the rate of the flux linkages and is expressed as follows [Ramu, 2001]:

$$V = RI + \frac{d\lambda(\theta, I)}{dt} = RI + \frac{\partial\lambda}{\partial I}\frac{dI}{dt} + \frac{\partial\lambda}{\partial\theta}\frac{d\theta}{dt} \qquad \text{Eq.(3-2)}$$

[4] Note that with the present test set-up (Appendix B), it is possible to control the *DC* and *FR* values individually for each motor phase to dictate motor performance in terms of lowering torque ripple or acoustic noise generation, but for simplicity, the same values are used for all phases during testing.

where V is the phase voltage, R is the phase resistance, I is the phase current, λ is the flux linkage per phase, and θ is the rotor angle[5].

The relation between flux linkage and current is nonlinear due to magnetic saturation. It also depends on the rotor position since the salient rotor and stator poles cause a variation in inductance of a phase (L) as the rotor rotates wrt to the stator (i.e. $L = f(\theta, I)$)

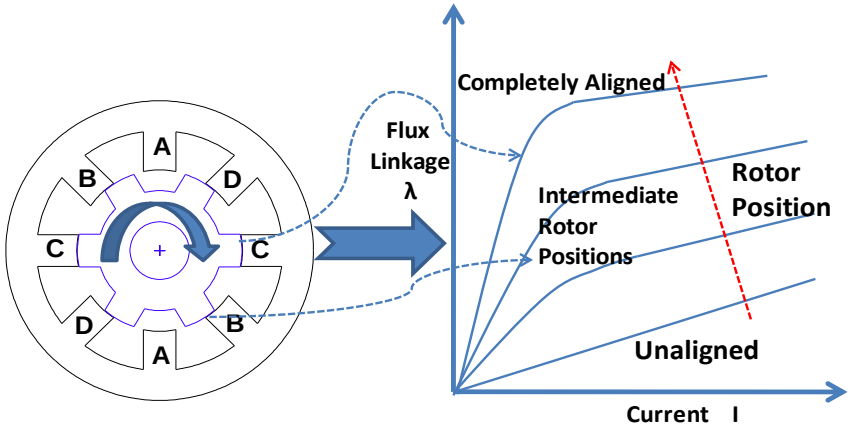


Figure 3-6: Flux Linkage Curves

The value of flux linkage is at its maximum when a rotor pole is completely aligned with a stator pole and is at a minimum when the rotor and stator poles are completely out of alignment (Figure 3-6). Assuming no mutual coupling between phases, the flux linkage can be represented as a set of curves of λ vs. I with θ as a parameter.

$$\lambda = L(\theta, I) \times I \qquad \text{Eq.(3-3)}$$

From the above equations, the voltage is given by:

$$V = RI + \left(L(\theta, I) + I \frac{\partial L(\theta, I)}{\partial I} \right) \frac{dI}{dt} + I\omega_m \frac{\partial L(\theta, I)}{\partial \theta} \qquad \text{Eq.(3-4)}$$

where $\omega_m = d\theta/dt$ is the motor speed. The three terms on the right hand side of Eq.(3-4) (which is, in essence, a simple nonlinear SRM model) represent the resistive voltage

[5] Typically, θ is assumed to have a value of zero when the rotor and the stator poles just begin to align and a value equal to the stator pole arc angle when the rotor and stator poles are completely aligned.

drop, the inductive voltage drop and the induced back emf respectively. When voltage is applied to the phase windings, the amplitude of the resulting current depends largely on the phase resistance. But the rate of rise or fall of current (*dI/dt*) depends on inductance as well as the back emf[6]. This, in turn, affects the torque produced. Typically the resistance does not vary much but in cases where the motor is pushed to its extreme performance limits, a change in the winding temperature (primarily due to I^2R or iron losses in the stator windings) causes a variation in resistance as given by:

$$R_t = R_0[1+\propto (T - T_0)] \qquad \text{Eq.(3-5)}$$

where R_t is the resistance at temperature T , R_0 is the resistance at temperature T_0, $\propto$ is the temperature coefficient of resistance. For the present work, this effect is not deemed to be significant and hence will not be included in the overall model. Thus, the influence of the phase voltage, phase resistance and the rotor angle on the phase current can be explained. The following paragraphs now explore the influence of the turn-on and turn-off angles for the phase on the current.

In a SRM, torque is produced by the tendency of the magnetic circuit to adopt a configuration where magnetic reluctance[7] to the flow of flux is minimal or the inductance is maximum both of which happen when the air gap between a stator pole and a rotor pole is minimum. For a particular phase, the stator magnetic field pulls the rotor and causes it to move until it aligns with the excited stator pole. In Figure 3-7, L_a and L_u represent the phase inductance values when the poles are completely aligned and when they are not aligned. By synchronizing the excitation of each phase with the rotor position, a continuous rotation of the rotor and hence a continuous torque in the desired direction can be obtained. For optimum torque production, ideally each phase must be switched on during the commutation sequence when its inductance starts to increase, and then switched off when the inductance starts to decrease. The rotor mechanical angles at

[6] This is similar to a typical R-L circuit with a resistor *R* and inductor *L* in series. With a constant voltage *V* applied to the circuit, the resulting current is given by $i = (V/R)(1 - e^{-(R/L)t})$, where *t* is the time.

[7] Reluctance in a magnetic circuit is analogous to resistance in an electrical circuit

which this switching happens are referred to as the 'turn-on' (θ_{on}) and 'turn-off' (θ_{off}) angles respectively (Figure 3-7).

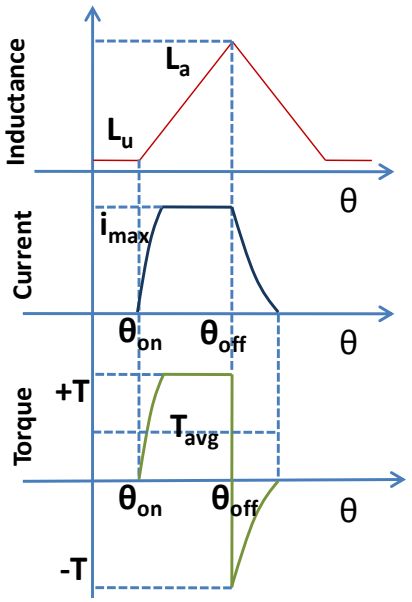


Figure 3-7: Inductance (idealized), Current and Torque profiles for a Single SRM phase

From the instant that a phase is switched on at θ_{on}, it takes some time for the current to build up to its maximum value. Similarly when the phase is switched off at θ_{off}, it takes some time for the current to fall to a zero value. This behavior is due to the inductance and the back emf in the circuit as mentioned earlier.

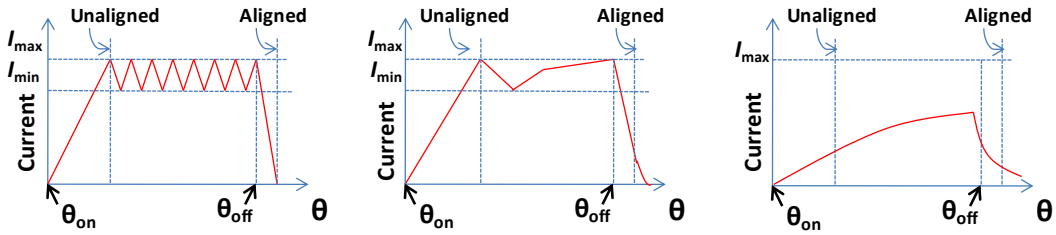
The resultant torque profile follows the current profile. Useful torque is produced only in the region where the inductance gradient is positive. But if the rotor continues past the completely aligned position (due to inertia), the attractive force from the stator magnetic field acting on it produces a braking effect. This negative torque produces no useful work. This effect is manifested as torque ripple, since the net torque produced by the SRM is a combination of torques produced by all the phases. Torque ripple contributes to vibration which in turn creates acoustic noise [Miller, 2002]. To mitigate this undesirable effect, it is necessary to switch off the current to a phase at an appropriate time before the stator and rotor poles start aligning or separating[8]. For a desired torque, if the current required is I_{max} and t_p is the time available for the current to reach I_{max}, then,

$$\int_0^{t_p} \frac{dI}{dt} dt \geq I_{max} \qquad \text{Eq.(3-6)}$$

[8] This procedure is referred to as 'current profiling' in the literature.

When the motor speed increases, the time available t_p decreases and it becomes difficult to regulate the current to its desired value. Thus, for optimum torque production, it is advantageous to switch on and switch off a phase in advance of θ_{on} and θ_{off} respectively, so the current has sufficient time to build up the flux from zero to the desired value before the inductance starts rising as well as sufficient time to decay to prevent producing any negative torque. Since the speed of the motor influences the back emf (Eq.(3-4)), there are different modes of performing this switching to control the current, based on the motor speed as described in [Panda and Amaratunga, 1993], [Ramu, 2001] and others.

When the SRM operates at a fairly low speed, the back emf value is also somewhat small as compared to the supply voltage. So, the current rises quickly to its desired value I_{max}. In this elapsed time, the angular motion of the rotor is relatively small. Hence it is necessary to switch the supply voltage (and thus the current) on/off rapidly in succession to maintain it at a constant value before the rotor pole is completely aligned with the stator pole. This is referred to as the 'chopping' mode of operation (Figure 3-8 (a)). In this mode, both θ_{on} and θ_{off} occur in advance of angles at which the stator and rotor poles are unaligned and fully aligned, respectively.



(a) Chopping Mode (b) Single Chop Mode (c) Single Pulse Mode

Figure 3-8: Phase Current Waveforms for Different Operating Modes

The next mode of operation is when the operating speed is moderately high (~60-80% of rated speed). The back emf is still not very significant [Panda and Amaratunga, 1993]. θ_{on} is advanced further ahead with respect to the unaligned position (more than in the chopping mode), and the rise time for the current from zero to I_{max} is relatively small.

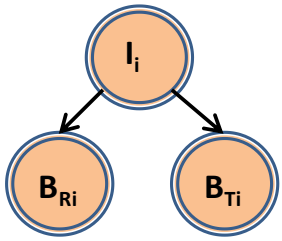
However, in the subsequent chop cycle, the rotor has already moved to a position of greater inductance since it is now more aligned with the stator pole. This results in a greater back emf which in turn causes the rate of rise of current to decrease. But the time for the current to rise to I_{max} is of the same order as the initial current rise time. Thus, there is only sufficient time for a single chop. This mode is therefore referred to as 'single-chop mode' (Figure 3-8 (b)).

The third mode of operation known as the 'single-pulse mode' (Figure 3-8 (c)) is applicable at high speeds (close to the rated motor speed). The back emf increases significantly, sometimes even exceeding the supply voltage. Since the rotor covers a greater angle in a given period, the inductance also increases rapidly. Both of these greatly diminish the rate of rise of the current and it is unable to reach a value of I_{max}. This in turn results in reduced torque output. In this mode, there is insufficient time for any chopping to take place and the motor operates with a single pulse of current in the phase winding. Thus, it is evident that in addition to factors like the phase voltage, the phase resistance and the rotor position, the values of θ_{on} and θ_{off} for a motor phase also play a significant role in determining the current drawn by that phase.

3.3.3. Relation between Current and Flux Densities

Often the standard equations that are used in the process of designing or controlling SRMs do not provide significant information on the radial and tangential force components in the air gap [Zhu et al., 2005]. An improved understanding of these force characteristics is however, crucial due to their influence on the overall torque and vibration characteristics of the motor. This section will examine the influence of the phase currents on the radial and the tangential flux density in the air gap between the stator and the rotor of the SRM. Figure 3-10 illustrates the general operating principle for one phase of the motor. When the stator windings are energized, it attracts the rotor pole nearest to it until the two are aligned. This attracting force $\vec{F}$ can be resolved into two components. The tangential component $\vec{F}_t$ is primarily responsible for producing electromagnetic torque on the rotor. The radial component $\vec{F}_r$ can be considered parasitic

in the sense that it contributes to the radial vibration of the stator poles by creating a deformation of the stator. This, in turn is the major contributor to acoustic noise in SRM drives [Pillay and Cai, 1999].



i = 1….n= Number of motor phases,

I_i= Phase Current

B_R= Magnetic Flux Density, radial

B_T = Magnetic Flux Density, tangential

Figure 3-9: Relation between Current and Flux Densities

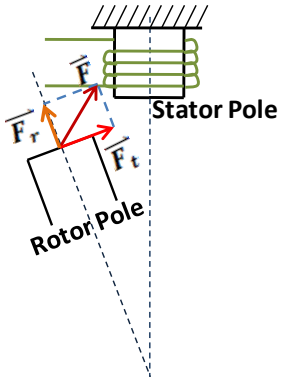


Figure 3-10 : SRM Principle

One of the methods used to numerically relate these two force components and their corresponding magnetic field components is the Maxwell Stress Tensor Method [Toth and Szabo, 2001][Benhama et. al., 1999]. This method requires the body under consideration to be located in the air or in a material with permeability $\mu = \mu_0$. The Maxwell stress tensor is defined as [Toth and Szabo, 2001]:

$$\underline{\vec{T}} = (\vec{B}.\vec{n})\vec{H} - \frac{1}{2}\mu H^2.\vec{n} \qquad \text{Eq.(3-7)}$$

where $\vec{n}$ is the unit vector normal to the surface. As described by [Toth and Szabo, 2001], the electromagnetic force ($\overrightarrow{F_{em}}$) is a combination of the Laplace force ($\vec{F}_L$) and the magnetic force ($\vec{F}_M$) and the electromagnetic impulse theorem provides the equivalence between $\overrightarrow{F_{em}}$ and the surface integration of $\underline{\vec{T}}$:

$$\overrightarrow{F_{em}} = \int_V (\vec{J} \times \vec{B})dV - \int_V \left(\frac{1}{2}H^2 grad\mu\right)dV = \vec{F}_L + \vec{F}_M = \int_S \underline{\vec{T}}d\vec{S}.\vec{n} \qquad \text{Eq.(3-8)}$$

where $\vec{J}$ is the current density[9], $\vec{B}$ is the total magnetic flux density ($\vec{F}_L$ is the result of the interaction between $\vec{J}$ and $\vec{B}$), $\vec{H}$ is the magnetic field intensity, μ is the permeability of iron (via the *B-H* curve), and *S* is the surface enclosing the body under the force.

The total magnetic flux density can be resolved into two orthogonal components: B_n normal to the surface *S* and B_t tangential to the boundary *S*. Thus, the flux density vector can be expressed as $\vec{B} = B_n.\vec{n} + B_t.\vec{t}$, where $\vec{t}$ and $\vec{n}$ are unit vectors in the tangential and normal directions respectively. If the total field acting on the body is known, then $\vec{F}_M$ (called $\vec{F}$ henceforth) can be determined using the above equations as follows [Toth and Szabo, 2001] [Benhama, 1999]:

$$\vec{F} = L\left[\int_S \frac{1}{\mu_0}(B_n B_t)dS\,.\vec{t} + \int_S \frac{1}{2\mu_0}(B_n^2 - B_t^2)\,dS.\vec{n}\right] \qquad \text{Eq.(3-9)}$$

where μ_0 is the permeability of free space and *L* is the length of the body normal to the surface *S*. Considering radial (normal) and tangential directions with respect to the motor poles, the above equation can be re-written as:

$$\vec{F} = L\left[\int_S \frac{1}{\mu_0}(B_r B_t)dS\,.\vec{t} + \int_S \frac{1}{2\mu_0}(B_r^2 - B_t^2)\,dS.\vec{r}\right] \qquad \text{Eq.(3-10)}$$

The terms in the above equation represent the radial and tangential force densities, $\overrightarrow{F_r}$ and $\overrightarrow{F_t}$ respectively as described in [Edrington et.al, 2003], where:

$$\overrightarrow{F_r} = \frac{1}{2\mu_0}\int_S (B_r^2 - B_t^2)\,dS.\vec{r} \qquad \text{Eq.(3-11)}$$

$$\overrightarrow{F_t} = \frac{1}{\mu_0}\int_S (B_r B_t)dS\,.\vec{t} \qquad \text{Eq.(3-12)}$$

Thus, from the above equations, it is clear that an understanding of the air gap flux density is crucial for optimizing not only the design but also the operation of a SRM, since, as described earlier; these force components determine the production of torque and acoustic noise.

[9] For a conductor with cross section area a_c carrying current *I*, the current density *J* is defined as $J = I / a_c$

Ideally, the flux density should be measured in real-time during the motor operation by using magnetic field sensors. The ideal location to obtain the magnetic flux density readings is in the air gap between the rotor and stator which is the area of the maximum flux density. However, the issue with such an arrangement is that the air gap dimensions are usually kept very small to maximize torque capability [Ramu, 2001] [Ashok and Tesar, 2002]. The sensor size must be suitable for integration in such tight spaces and should not mechanically interfere with the rotor rotation. The sensor should also not affect the existing flux density distribution with its presence. Although there are sensors available which satisfy these constraints (for example, miniature transverse Hall-generators from Lakeshore Cryotronics [Yoo and Tesar, 2004]), unless their inclusion is accounted for in the design phase of the motor, it is often difficult to accommodate them after the motor has been manufactured without significantly affecting the motor performance characteristics. Hence, the flux distribution is often obtained analytically or numerically via finite element techniques with the knowledge of other variables.

The analytical methods rely on the calculation of the aligned and unaligned magnetization curves and the use of special numerical functions to represent the magnetization curves (flux linkage versus position with current as a parameter) as discussed in [Matveev, 2006] [Miller and McGilp, 1993], etc. Although they are faster for simulation in many cases, the initial problem set-up procedure is often laborious [Deihimi et. al. 2002] and requires interpolation to calculate the individual magnetization curves between the aligned and unaligned positions[10]. They also do not provide much information on the distribution of the radial and tangential components of the flux [Edrington, 2003] [Parreira et. al., 2005]. In finite element analysis (FEA), the problem domain is discretized into small elements and numerical techniques are used to obtain approximate solutions to boundary-value problems. It is well suited for problems with complex magnetic circuits, nonlinear material properties, etc. and hence has acquired

[10] Analytical models used for an SRM are typically based only on the voltage equation ($V = RI + \frac{d\lambda(\theta,I)}{dt}$) and the torque equation ($T = \frac{dW'}{d\theta}|_{\text{i=const}}$ where W' is the co-energy) and do not take into account the magnetic and electric coupling of the motor phases [Matveev, et. al., 2003].

wide acceptance in modeling/analysis of SRMs where modeling issues arise primarily due to the double salient structure and magnetic saturation effects that occur in the partially aligned stator–rotor poles [Mizia et. al, 1988][Ramu, 2001][Miller, 2002][Lin, et. al. 1996][Omekanda,et. al., 1996]. [Matveev, 2006] demonstrates with example calculations of the static magnetization and torque curves for different SRMs, using both the analytical and FEA procedures (2-D), that the analytical results are less accurate and reliable than those obtained by FEA when the end-effects are not taken into account.

The underlying principle of FEA for electromagnetic systems is minimization of the energy functional *F*, the difference between input and the stored energy [Matveev, 2006]

$$F = \int_V \left(\int_0^B \vec{H} d\vec{B} - \int_0^A \vec{J} d\vec{A} \right) dV \qquad \text{Eq.(3-13)}$$

where, $\vec{A}$ is the magnetic vector potential. The two classical FEA methods to formulate this problem are the Ritz variation method and the Galerkin method [Matveev, 2006]. Once the problem is formulated, the subsequent step involves solving a system of equations. Most commercially available FEA software (e.g. ANSYS) use Maxwell's differential equations as the basis for simulating and analyzing electromagnetic fields. The subset of Maxwell's equations relevant to the present discussion are the Maxwell-Ampere's law, Faraday's law and Gauss's law for magnetism, as listed below.

$$\nabla \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t} \qquad \text{Eq.(3-14)}$$

$$\nabla \times \vec{\mathrm{E}} = -\frac{\partial \vec{\mathrm{B}}}{\partial \mathrm{t}} \qquad \text{Eq.(3-15)}$$

$$\nabla . \vec{B} = 0 \qquad \text{Eq.(3-16)}$$

where $\vec{D}$ is the electric flux density vector and $\vec{\mathrm{E}}$ is the electric field intensity vector.

Ideally any FEA should be done in three dimensions to represent the complete geometry of the electromagnetic structure being analyzed. But such models are more complex to generate than 2D models. Hence, most FEA involves 2D analysis by making

some assumptions to arrive at a reliable model as described in [Ji, et. al., 2005]. The magnetic field distribution is assumed to be constant along the motor axis. The stator and rotor materials are assumed to be isotropic and hysteresis effects are neglected. The outer periphery of the stator is considered the zero magnetic vector potential line due to the negligible field outside the stator. The currents induced in the stator and rotor iron are neglected due to the high resistance of the steel laminations to the eddy currents along the motor axis. [Matveev, 2006] demonstrates that the magnetic parameters of a SRM can be derived using such 2D analysis and then corrected for end-effects analytically without significant loss of accuracy. With these assumptions, the above equations now become:

$$\nabla \times \vec{\mathrm{H}} = \vec{\mathrm{J}}\,,\ \nabla \times \vec{\mathrm{E}} = 0 \text{ and } \nabla.\vec{B} = 0$$

A solution for these equations can be obtained using potential functions. Based on factors like field dimensionality, dynamics, domain size, discretization, etc. the available choices are the magnetic vector and scalar potentials [Matveev, 2006]. As magnetostatic fields are non-conservative, a scalar potential is unsuitable. Using a magnetic vector potential $\vec{A}$, the magnetic field can be obtained by derivatives. For any vector $\vec{A}$,

$$\nabla.(\nabla \times \vec{A}) = 0 \qquad \text{Eq.(3-17)}$$

$$\nabla \times (\nabla \times \vec{A}) = \nabla(\nabla.\vec{A}) - \nabla^2\vec{A} \qquad \text{Eq.(3-18)}$$

To completely and uniquely specify $\vec{A}$, it is necessary to specify its curl and its divergence (Helmholtz theorem). This can be done by using the Coulomb gauge equation for magnetic vector potential [Stiles, 2007]

$$\nabla.\vec{A} = 0 \qquad \text{Eq.(3-19)}$$

$$\nabla \times (\nabla \times \vec{A}) = -\nabla^2\vec{A} \qquad \text{Eq.(3-20)}$$

$$\overrightarrow{\therefore B} = \nabla \times \vec{A} \qquad \text{Eq.(3-21)}$$

By definition, $\vec{B} = \mu\vec{H} = \mu_0\mu_r\vec{H}$ where μ is the magnetic permeability of the iron, μ_0 is the absolute permeability, μ_r is the relative permeability. Thus,

$$\nabla \times \left(\frac{1}{\mu_0 \mu_r} . \nabla \times \vec{A}\right) = \vec{J} \qquad \text{Eq.(3-22)}$$

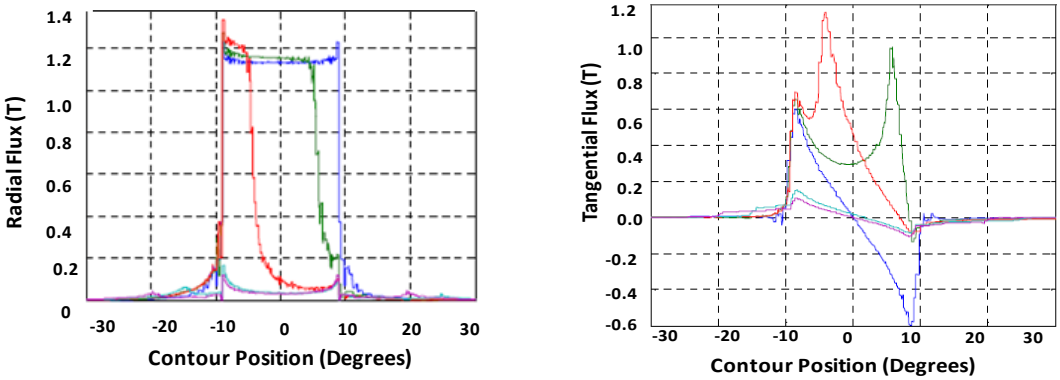
$$\therefore -\nabla^2 \vec{A} = \vec{J} \qquad \text{Eq.(3-23)}$$

Considering the 2D (planar) static magnetic analysis, the components A_x and A_y of $\vec{A}$ are not applicable. The only relevant component of current density $\vec{J}$ is J_z (or J, henceforth) and the only relevant component of $\vec{A}$ is A_z (or A, henceforth). Thus,

$$\frac{\partial}{\partial x}\left(\frac{1}{\mu_0 \mu_r}\frac{\partial \mathrm{A}}{\partial \mathrm{x}}\right) + \frac{\partial}{\partial y}\left(\frac{1}{\mu_0 \mu_r}\frac{\partial \mathrm{A}}{\partial \mathrm{y}}\right) = -J \qquad \text{Eq.(3-24)}$$

The above equation is referred to as the Poisson's equation for the planar case with a perpendicular current carrying conductor and can be solved using the Dirichlet and Neuman boundary conditions for the magnetic vector potential [Matveev, 2006]. The two components of the flux density in the above case are defined as:

$$B_x = \frac{\partial A}{\partial y} \text{ and } B_y = \frac{\partial A}{\partial x} \qquad \text{Eq.(3-25)}$$



Radial Magnetic Field | Tangential Magnetic Field

Rotor Position Unaligned To Aligned, i=30A

[Edrington et. al., 2003]

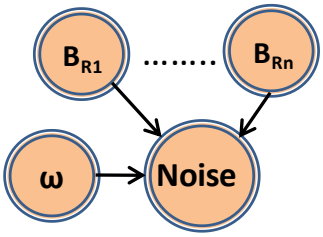
Figure 3-11: Variation of Radial and Tangential Components of Magnetic Flux Density

The combined effect of all such components, taken at each rotor position and over the length of the stator stack, result in the net tangential and radial components

respectively. From the above equations, it can be concluded that the current density in the phase coils directly influences the magnetic vector potential, the gradients of which, in turn, determine the radial and the tangential components of the magnetic flux density. Figure 3-11 shows a typical variation of the field components in the air gap of a 8/6 SRM for a current of 30A as the rotor moves into alignment with the stator pole from a completely unaligned position [Edrington, et. al. 2003].

3.3.4. Relation between Flux Density, Speed and Acoustic Noise

The sources of acoustic noise in a SRM can be broadly categorized into mechanical, aerodynamic, electronic and magnetic sources. [Vijayaraghavan and Krishnan, 1999] and [Pillay and Cai, 1999] provide in-depth analyses of these sources of noise as well as control techniques that can be used to mitigate the generation of excessive noise. The mechanical sources include factors like the coupling between the motor and its load, the mounting method used for the motor, noise arising from the motor bearings (due to the bearing type, installation, etc.), rotor imbalance, etc. A non-uniform air gap between the rotor and stator due to manufacturing inaccuracies or non-isotropic rotor and stator materials result in the creation of uneven magnetic flux. This, in turn, creates asymmetric forces on the both the rotor and stator producing vibrations that are manifested as acoustic noise. The aerodynamic sources of noise primarily arise from windage effects. The doubly salient structure of a SRM causes the rotor poles to act as blades as they rotate in the air, causing acoustic noise. This category also includes noise arising from the motor's cooling fans. The electronic sources of noise arise primarily from the motor's drive system. SRMs are inherently prone to produce more noise than smooth air-



i = 1….n = Number of motor phases,

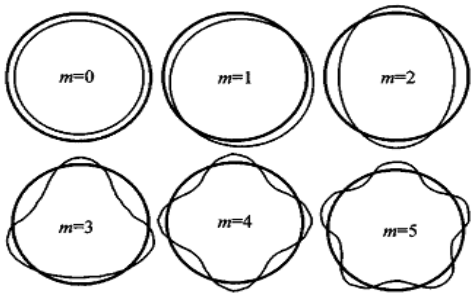
B_R= Magnetic Flux Density, radial

ω = Motor Speed, N= Acoustic Noise

Figure 3-12 : Relation between Radial Flux Density, Motor Speed and Acoustic Noise

gap machines due to the fact they are operated with pulsed excitation [Vijayaraghavan and Krishnan, 1999]. The main factors that contribute towards this are the presence of harmonics in the applied voltage and current. The currents through the stator windings interact with the local magnetic field to produce forces on the windings, which can create winding vibrations. The vibrations induced are typically of higher amplitudes at higher currents or when the poles are close to alignment, or when the rate of change of current is high [Vijayaraghavan and Krishnan, 1999].

As seen earlier, torque in a SRM is produced due to the attractive force between an excited stator phase and the rotor created by the magnetic field. This force contains a radial component in addition to the tangential component that produces useful torque. The magnetic sources of noise in the SRM are primarily due to these radial forces. A variation in the flux density as the rotor aligns with the stator poles, results in a variation in the corresponding radial force on the stator. This induces deformations of the stator lamination stack. The torque ripple[11] in the torque profile of a SRM (produced as a result of commonly used commutation/control strategies and drive electronics) can be directly correlated with the tangential magnetic force acting on the motor poles. To a lesser extent, this also contributes to stator vibrations. The stiffer rotor is not deformed as much and its reaction to most forces is small [Wu and Pollock, 1999]. If a certain radial force frequency coincides with the natural frequency of the stator stack, then resonant vibrations occur [Colby et. al., 1995], manifested in various mode shapes.



m=mode number

[Anwar and Hussain, 2000]

Figure 3-13: Circumferential Vibration Modes

The natural frequencies of these mode shapes depend on the motor geometry and its material properties. The circumferential modes of vibration are more

[11] The torque ripple can be defined as $T_{ripple} = |(T_{max}-T_{min})/T_{avg}|$ [Wichert, 2008]. $T_{max}, T_{min}, T_{avg}$ are the maximum, minimum and average value of the motor torque

dominant in their contribution to acoustic noise than the longitudinal ones [Anwar and Hussain, 2000]. For airborne noise, the significant modes of vibrations for small machines are 1-4. [Colby et. al., 1995] demonstrate that the intensity of acoustic noise is most when the second order circumferential mode (m=2) is excited. [Anwar and Hussain, 2000] provide an analytical expression for the amplitude of dynamic circumferential deflection for modes >= 2 as:

$$D_{circum} = \frac{\frac{12F_r(f_{exc})R_m}{m^4E}\left(\frac{R_m}{h_s}\right)^3}{\sqrt{\left\{1-\left(\frac{f_{exc}}{f_m}\right)^2\right\}^2+\left(\frac{\delta}{\pi}\frac{f_{exc}}{f_m}\right)^2}} \qquad \text{Eq.(3-26)}$$

where, D_{circum} is the amplitude of dynamic deflection, F_r is the amplitude of the radial force wave (radial force/unit operating area), f_{exc} is the excitation frequency, m is the circumferential mode number, f_m is the mode frequency, R_m is the mean radius of the stator yoke, h_s is the stator pole height, and δ is the logarithmic decrement ($= 2 \times \pi \times$ damping factor). The magnitude of acoustic noise depends on the extent of D_{circum}. The sound power radiated by the motor is then given as [Anwar and Hussain, 2000]:

$$P = 4.\sigma_{rel}.\rho.c.\pi^3.f_{exc}^3.D_{circum}^2.R_{out}.l_{stk} \qquad \text{Eq.(3-27)}$$

where P is the sound power radiated, σ_{rel} is the relative sound intensity = k^2 / (1+ k^2) , k is the wave number = (2. π. R_{out}. f_{exc})/c, c is the speed of sound in the medium, ρ is the density of the medium, R_{out} is the outer radius of the stator, and l_{stk} is the stator stack length. Assuming a reference sound power level P_{ref} of 10^{-12} watt (the threshold of human hearing), the acoustic noise power in decibels is given by [Anwar and Hussain, 2000]:

$$L_w = 10\,log\left(\frac{2.P}{P_{ref}}\right) \qquad \text{Eq.(3-28)}$$

For analyzing vibrations, a SRM can be represented as a linear superposition of multiple single degree of freedom (dof) viscously damped spring-mass systems, although it is a multi-dof system [Cai et. al., 2001]. The stator vibration can be expressed as [Michaelides and Pollock, 1999]:

$$F_r = m\frac{d^2x}{dt^2} + b\frac{dx}{dt} + kx \qquad \text{Eq.(3-29)}$$

where F_r is the radial force, m is the equivalent mass, x is the stator displacement, b is the viscous damping coefficient and k is the stiffness. Re-writing the above equation,

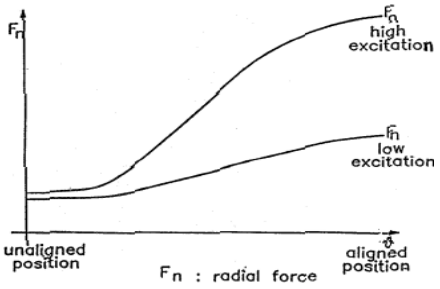
$$\frac{F_r}{m} = \frac{d^2x}{dt^2} + 2\xi\omega_n\frac{dx}{dt} + \omega_n^2 x \qquad \text{Eq.(3-30)}$$

where $\xi = \frac{b}{2\sqrt{km}}$ is the damping ratio and $\omega_n = \sqrt{\frac{k}{m}}$

Typically, the structural vibrations are measured using accelerometers. If a is the acceleration corresponding to the displacement x, then differentiating the above equation twice and simplifying [Michaelides and Pollock, 1999],

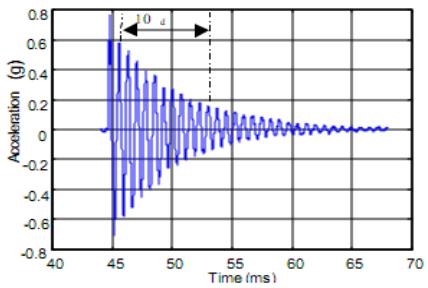
$$\frac{1}{m}\frac{d^2F_r}{dt^2} = \frac{d^2a}{dt^2} + 2\xi\omega_n\frac{da}{dt} + \omega_n^2 a \qquad \text{Eq.(3-31)}$$

From Eq.(3-31), it can be seen that the second derivative of the radial force F_r drives the stator acceleration and thus it is important to understand the factors influencing it [Michaelides and Pollock, 1999].



[Michaelides and Pollock, 1999]

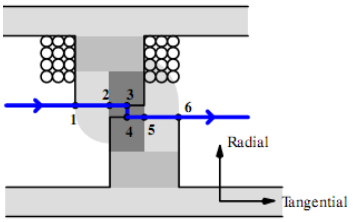
Figure 3-14: Variation of Radial Force in a SRM



[Cai, et. al., 2001]

Figure 3-15: SRM Vibration Acceleration

The radial force may be calculated using FEA. Analytically, the static radial force characteristics maybe determined using energy based methods as described in [Ramu, 2001] or alternatively, by the Maxwell stress tensor. From Eq.(3-11), it can be seen that



[Garrigan, et. al, 1999]

Figure 3-16: Using Maxwell' Stress Tensor to determine Radial Force

F_r depends on both the tangential as well as the radial components of the magnetic flux density. The equation also provides a method to estimate the value of F_r analytically. By selecting an appropriate integration path around the individual motor poles and assuming that flux paths can be approximated as being purely radial across the air gap, [Garrigan, et. al., 1999] illustrate this with a simple example. The contour selected for stress calculation is such that the flux lines are either orthogonal or parallel to the contour. This yields forces only normal to the contour and are given by simplifying Eq.(3-11) [Garrigan, et. al., 1999]:

For orthogonal field lines: $$F_N = \frac{1}{2\mu_0}\int_S (B_r^2)\, dArea \qquad \text{Eq.(3-32)}$$

For parallel field lines: $$F_N = \frac{1}{2\mu_0}\int_S (-B_t^2)\, dArea \qquad \text{Eq.(3-33)}$$

If L_{stack} is the stack or the iron length along the motor axis, B_m is the main air gap magnetic field (primarily along the radial direction when the poles begin to align) and B_{f1} and B_{f2} are the first and second fringing fields, then using the above equations, for the contour shown in Figure 3-16, F_r can be calculated as follows [Garrigan, et. al., 1999]:

$$F_r = \frac{L_{stack}}{2\mu_0}\left(\int_1^2 B_{f1}^2\, dl + \int_2^3 B_m^2\, dl + \int_4^5 B_m^2\, dl + \int_5^6 B_{f2}^2\, dl\right) \qquad \text{Eq.(3-34)}$$

$$= \frac{L_{stack}}{2\mu_0}\left(B_{f1}^2 l_{12} + B_m^2(l_{23} + l_{45}) + B_{f2}^2 l_{56}\right) \equiv \frac{B_r^2\, Area}{2\mu_0}$$

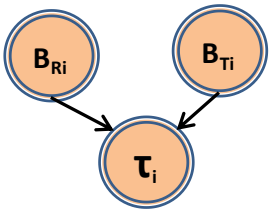
Thus, the radial force can be considered to be equivalent to integrating the radial flux density B_r over the surface area of the air gap [Garrigan, et. al., 1999][12], since all the

[12] This is only an approximation since B_t also influences F_r but not as much as B_r

other parameters in the above equation are fixed by the motor geometry or are constants. The net force is obtained by combining all such forces vectorially.

3.3.5. Relation between Flux Density and Torque

As seen in the earlier sections, all the major electromagnetic parameters for a SRM like the phase inductance, current drawn by a phase, flux density, etc. vary continuously with the rotor position due to the doubly salient geometry of the motor. The phase current influences the overall air gap magnetic field and the tangential component of the force produced by this field creates the useful motor torque. Thus the torque produced by the motor also continuously varies as the rotor poles move from the unaligned to the aligned position with respect to the stator poles. The two popular methods to determine the SRM static torque are by using the Maxwell stress tensor or via the virtual work principle. Using the 2-D Maxwell stress tensor formulation as described earlier, the torque can be calculated as [Mizia, et. al., 1988][Moallem et. al., 1992]:



i = 1….n = Number of motor phases,

B_R= Magnetic Flux Density, radial

B_T = Magnetic Flux Density, tangential

τ_i = Torque produced by a single phase

Figure 3-17: Relation between Flux Density and Torque

$$\tau = \frac{1}{\mu_0} L_{stack} \int_{\Gamma} [(r B_r B_t)] d\Gamma \qquad \text{Eq.(3-35)}$$

where L_{stack} is the stack length of the iron core, Γ is the closed contour around the stator, and r is the radius of the cylindrical surface in the mid air gap around which the integration is performed. Using the finite element formulation, the above equation becomes:

$$\tau = \frac{1}{\mu_0} L_{stack} r^2 \sum_{i=1}^{m} B_{ri} B_{ti} dt_i \qquad \text{Eq.(3-36)}$$

where m are the discrete position values at which the magnetic field is measured [McCann, et.al, 2008]. The above expression clearly shows the influence of the product of the two flux densities on the motor torque.

[Moallem, et. al, 1992] demonstrate that the radial component of the flux density is dominant at most of the overlapping positions between the rotor and the stator poles. The tangential component, on the other hand, does not have a very significant effect except around the pole tips of the excited stator pole and the nearest rotor poles. The authors also investigate the extent that certain rotor pole profiles have in shaping the tangential flux on the distribution on the pole face and their impact on the torque produced and conclude that decreasing the air gap length will result in an increase in the magnitude of B_r and slanting or serrating the rotor pole phase influences the profile of B_t, both of which in turn significantly affect the average torque. Thus, due to such close coupling between the two components of the flux density, it is more beneficial to examine the effect of overall air gap flux density on the torque. [Garrigan et. al, 1999] again provide a simple expression to approximately determine the torque based on a judicious choice of integration contour (Figure 3-16):

$$\tau = \frac{rL_{stack}}{2\mu_0}\int_3^4 [(-B_m^2)]d\Gamma = \frac{-rL_{stack}B_m^2 l_{34}}{2\mu_0} \qquad \text{Eq.(3-37)}$$

where l_{34} represents the air-gap length. The above expression is based on determining the value of B_m using the Maxwell's stress tensor formulation.

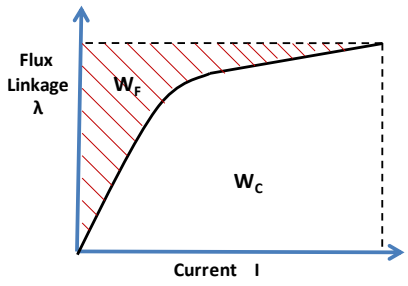


Figure 3-18: Stored Field Energy and Co-energy

However, the more commonly adopted method to determine the torque is based on the virtual work principle since the variables that influence the torque equation are obtainable with relative ease (measured by sensors or determined analytically with a high degree of accuracy). The SRM magnetization curves form the basis for torque calculations using this method. In this approach, torque is calculated by evaluating

the rate of change of co-energy (defined as the complement of the energy stored in the air gap field [Ramu, 2001]) with respect to the rotor angular displacement. The area above the magnetization curve represents the stored energy (W_F) and that below the curve represents the co-energy (W_c) (Figure 3-18). The energy balance can thus be given by:

$$W_F + W_c = I\lambda \qquad \text{Eq.(3-38)}$$

Multiplying Eq.(3-2) by current *I* and simplifying,

$$VI\,dt = I^2Rdt + I\frac{\partial\lambda}{\partial I}dI + I\frac{\partial\lambda}{\partial\theta}d\theta \qquad \text{Eq.(3-39)}$$

The product term on the left hand side of the above equation is the rate of change in the supplied electrical input energy (W_S). The terms on the right hand side represent the rate of energy dissipated as copper losses (W_L), the change in the energy stored in the magnetic field (W_F) and the change in the energy converted to mechanical work (W_M) [Wichert, 2008]. The difference between the input and the dissipated energy is given by

$$dW_s - dW_L = dW_E = dW_F + dW_M = I\frac{\partial\lambda}{\partial I}dI + I\frac{\partial\lambda}{\partial\theta}d\theta \qquad \text{Eq.(3-40)}$$

Due to the double saliency of the SRM, the incremental energy stored in the field is a function of both the rotor position θ and current *I,* i.e. $W_F = f(I, \theta)$ [Wichert, 2008].

$$dW_F = \frac{\partial W_F}{\partial I}dI + \frac{\partial W_F}{\partial\theta}d\theta \qquad \text{Eq.(3-41)}$$

$$dW_M = \left(I\frac{\partial\lambda}{\partial I} - \frac{\partial W_F}{\partial I}\right)dI + \left(I\frac{\partial\lambda}{\partial\theta} - \frac{\partial W_F}{\partial\theta}\right)d\theta \qquad \text{Eq.(3-42)}$$

The energy stored in the air gap field for a given rotor position (Figure 3-18) is given by

$$W_F = \int_0^{\lambda} I\,d\lambda = I\lambda - \int_0^{I}\lambda\,dI \qquad \text{Eq.(3-43)}$$

Taking a partial derivative w.r.t *I*,

$$\frac{\partial W_F}{\partial I} = I\frac{\partial\lambda}{\partial I} + \lambda - \lambda(1) = I\frac{\partial\lambda}{\partial I} \qquad \text{Eq.(3-44)}$$

$$\therefore dW_M = \left(I\frac{\partial\lambda}{\partial\theta} - \frac{\partial W_F}{\partial\theta}\right)d\theta$$

The rate of change of mechanical work can be expressed in terms of the torque produced by each phase and the differential change in the rotor position as:

$$dW_M = \tau_{ph}d\theta \quad \text{Eq.(3-45)}$$

$$\tau_{ph} = I\frac{\partial\lambda}{\partial\theta} - \frac{\partial W_F}{\partial\theta} \quad \text{Eq.(3-46)}$$

From Eq.(3-38), for a constant value of *I,* partially differentiating wrt θ

$$\frac{\partial W_F}{\partial\theta} + \frac{\partial W_C}{\partial\theta} = I\frac{\partial\lambda}{\partial\theta} \quad \text{Eq.(3-47)}$$

Thus, the instantaneous torque produced by each phase is given by:

$$\tau_{ph} = \frac{\partial W_c}{\partial\theta}\Big|_{I=const} \quad \text{Eq.(3-48)}$$

The total instantaneous motor torque is the sum of the torques produced by each phase:

$$\tau_{inst} = \sum_{i=1}^{no.of\ phases} \tau_{ph} \quad \text{Eq.(3-49)}$$

A continuous torque is obtained by sequentially switching the different motor phases in order (the unequal number of stator and rotor poles in a SRM ensures that at every instant, there is at least one pair of unaligned poles which can be energized to create a rotation as shown in Figure 3-10). In order to obtain sufficient accuracy for real-time applications (due to the presence of the derivative, the estimates may be fluctuating), torque is often computed over a time period as an average value [McCann, et al 2008]:

$$\tau_{avg} = \frac{1}{T}\int_0^T \tau_{inst}\,dt \quad \text{Eq.(3-50)}$$

Thus, the torque can be determined from the co-energy for a given current value.

[Ramu, 2001] provides a method to determine the value of co-energy. Assuming the iron to be infinitely permeable, the only reluctance in the circuit is provided by the air gap between the stator and the rotor. The total air gap flux density *B* can be expressed

purely as a function of the pole angle of overlap between the stator and the rotor i.e. θ, the current I, and the air gap dimension g as follows [Ramu, 2001]:

$$B = \frac{\phi}{A} = \frac{\phi}{L_{stack} r\theta} = \mu_0 H = \mu_0 \frac{NI}{g} \qquad \text{Eq.(3-51)}$$

where ϕ is the flux, N is the number of turns in the phase coil (λ=Nϕ), H is the magnetic field strength, L_{stack} is the iron length (along the motor axis), r is the rotor radius, and μ_0 is the permeability of air. Re-arranging, the flux-linkage is given by:

$$\lambda = \frac{N^2 \mu_0 L_{stack} r\theta I}{g} \qquad \text{Eq.(3-52)}$$

The co-energy W_c is then calculated as:

$$W_c = \int_0^I \lambda \, dI = \frac{1}{2} \frac{g\lambda^2/N^2}{\mu_0 L_{stack} r\theta} = \frac{1}{2} \frac{g\phi^2}{\mu_0 L_{stack} r\theta} = \frac{1}{2} \frac{gB^2 r\theta L_{stack}}{\mu_0} \qquad \text{Eq.(3-53)}$$

Differentiating partially wrt to θ, for a constant current I,

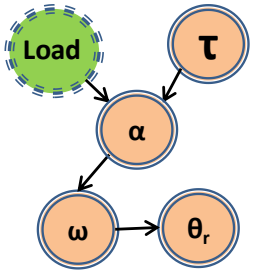
$$\left.\frac{\partial W_c}{\partial \theta}\right|_{I=const} = -\frac{1}{2} \frac{g\phi^2}{\mu_0 L_{stack} r\theta^2} = \frac{1}{2} \frac{gB^2 r L_{stack}}{\mu_0} = \tau_{ph} \qquad \text{Eq.(3-54)}$$

The above expression is similar to the one proposed by [Garrigan et. al, 1999]. Thus it can be seen that, given the geometrical parameters of the motor, the torque produced by any of the motor phases is primarily determined by the magnitude of the magnetic flux density in the air gap.

3.3.6. Relation between Motor Torque, Acceleration, Speed and Position

The torque generated by the motor magnetic field acts against the friction in the system (arising from bearings, seals, etc.) and any external load acting on the motor shaft[13] in order to propel the rotor inertia. The resultant acceleration, speed and position of the motor shaft are thus a direct outcome of the torque generation mechanism. The

[13] In the present case, the load is determined by a hysteresis brake attached to the motor. When the motor is coupled to a geartrain, the gear ratio dictates the load reflected back to the motor. Also, only viscous friction will be considered in the present case. The method proposed in [Gobbi, et. al., 2007] may be adopted to include the Coulomb friction effects in the dynamic model for the SRM.



τ = Total Electromagnetic Motor Torque

α = Angular Acceleration of Rotor

ω = Angular Velocity of Rotor

θ_r = Rotor Position

Load = External Load Torque

Figure 3-19: SRM Dynamics

dynamic behavior of the SRM can be represented using standard equations of motion as follows:

$$\omega = \frac{d\theta_r}{dt} \qquad \text{Eq.(3-55)}$$

$$\alpha = \frac{d\omega}{dt} \qquad \text{Eq.(3-56)}$$

$$\tau_e - \tau_L = J\alpha + A\omega \qquad \text{Eq.(3-57)}$$

where $\tau_e = \sum_{i=1}^{no.of\ phases} \tau_{ph}$

where θ_r is the rotor position, ω represents the rotor angular velocity, α is the angular acceleration, J is the combined inertia of the rotor and the load, A is the coefficient of viscous damping, τ_e is the total electromagnetic torque produced by all the motor phases combined and τ_L is the external load torque acting on the motor shaft.

3.3.7. Power Conversion and Efficiency of a SRM

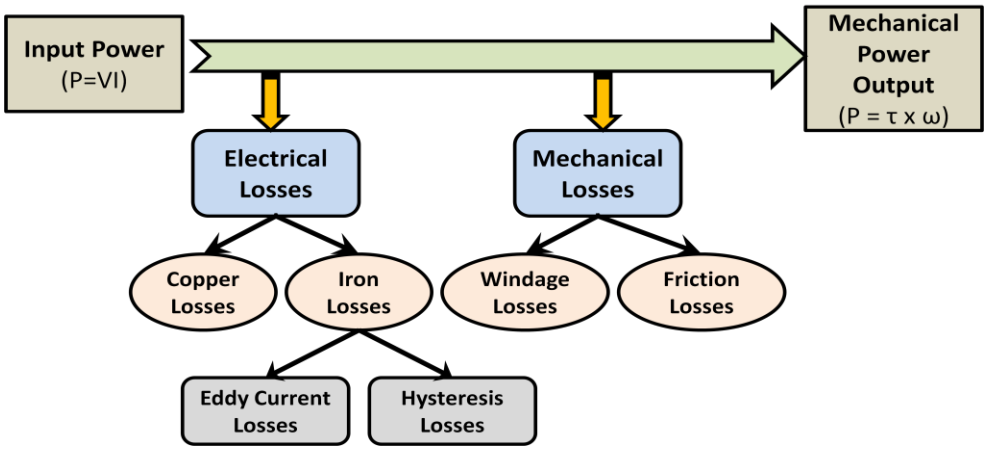


Figure 3-20: Power Losses in a SRM

In any motor, ideally, all the electrical input power (P_{elec} = VI) would be converted completely into useful output mechanical power ($P_{mech} = \tau\omega$). However, in a real system, different types of internal losses, both electrical and mechanical, occur during the process of this energy conversion and affect the overall motor efficiency[14]. To alleviate the effects of such losses, there is a need for a better understanding of the factors causing them. These include design choices like the type of steel laminations used in the motor construction [Hasegawa et al., 2002], winding configurations, etc. as well as operational factors like speed, the PWM parameters, etc [Ashok and Tesar, 2002]. The typical losses in a SRM are as follows:

***Electrical losses*:** This category includes two main types of losses, namely copper or Ohmic losses and iron or core losses. Copper losses are the consequence of the resistance of the motor phase windings to the current flow and are given by [Miller,1993]:

$$P_{Cu} = nI^2R_{ph} \qquad \text{Eq.(3-58)}$$

where n represents the number of motor phases, I is the rms value of the phase current, and R_{ph} is the resistance per phase (assumed to be approximately the same for all the motor phases). The phase resistance is affected by a number of factors like temperature, skin effect (uneven flow of current across the cross section of a conductor), proximity effect (caused by the interaction of magnetic fields in adjacent conductors), etc., all of which eventually influence the current density and usually have an adverse effect on the copper losses [Ashok and Tesar, 2002].

The losses may be particularly highly at high excitation levels when the motor is being pushed to its limits and draws high currents. In order to estimate the magnitude of these losses accurately, the exact current profile needs to be known but this is often complicated given the non-sinusoidal nature of the current in a SRM and its dependence on the motor speed and the switching strategy being implemented. [Materu and Krishnan, 1992] suggest using an approximation that assumes an ideal current waveform with zero

[14] To accurately determine the overall system efficiency, it is also essential to account for additional losses that may exist in the motor power supply, conduction and switching losses in the motor controller, the gear train efficiency (when the motor and the geartrain are integrated in an EMA), etc.

rise time and a flat peak value of I_p to estimate the rms value of the phase current as $I = I_p/\sqrt{n}$, which in turn can be used to estimate the value of copper losses.

The other major component of the electrical losses are known as core losses which occur due to currents induced in the ferromagnetic motor material as well as due to the characteristic response of the material to applied magnetic fields. These two constituents are referred to as the eddy current loss and hysteresis loss respectively. Under a time varying magnetic field, a voltage is induced in ferromagnetic cores (the stator iron stack in case of SRMs), causing eddy currents which flow in closed paths. The direction of flow of these currents is such that it opposes the rate of change of flux that would be caused by the voltage source[15]. Hysteresis losses represent the rate of change of the energy used to affect magnetic domain wall motion as the domain grows and rotates under the influence of an externally applied magnetic field [Raulin, et. al., 2004]. These losses hamper the rapid buildup of flux in the motor air gap which affects the speed of response. They are independent of the motor load but depend on the switching frequency of the applied voltage. [Miller, 1993] states that although the magnitude of total core losses in SRMs is relatively low as compared to other motor types, they are dominant at higher speeds relative to the other losses. The two components of the core losses can be determined using the Steinmetz equation described in [Miller, 1993] as follows:

$$P_{Fe} = P_{ec} + P_h = C_e f^2 B_{max}^2 + C_h f B_{max}^{a+bB_{max}} \qquad \text{Eq.(3-59)}$$

where P_{Fe} represents the total core losses, P_{ec} represents the eddy current losses, P_h represents the hysteresis losses, C_e and C_h are the Steinmetz coefficients for the eddy-current and hysteresis losses respectively, $B_{\max}$ is the peak magnetic flux density, f is the switching frequency and a, b are constants for the material of the electromagnetic core (usually derived experimentally). A limitation of this approach is that the use of the Steinmetz equation requires the flux waveforms to be sinusoidal, which is not the case in SRMs. [Miller, 1993] suggests a modification of the eddy-current term in the above equation in order to account for the non-sinusoidal flux waveforms as:

[15] The stator of the SRM is built up with laminations to reduce the effect of eddy currents.

$$P_{ec} = C_{e1}\left[\frac{dB}{dt}\right]^2 = \frac{C_e}{2\pi^2}\left[\frac{dB}{dt}\right]^2 \qquad \text{Eq.(3-60)}$$

***Mechanical Losses*:** Mechanical losses are a significant component of the total losses in a SRM. The two types of losses included in this category are friction losses and windage losses**.** Friction losses may be either Coulomb losses (e.g. in shaft seals) or viscous friction losses, primarily occurring in the motor bearings, due to the effect of motor operating speed on lubrication and material damping characteristics. [Dabala, 2001] provides an approximation for friction losses (in *W*, for a single bearing) as:

$$P_{mbe} = \frac{1.5\,F\,v}{d_{be}}10^{-5} \qquad \text{Eq.(3-61)}$$

where *F* is the radial force (in *N*) on the bearing, v is the perimeter speed on the bearing race surface (m/s) and d_{be} is the average diameter of the roller elements (m). [Dabala, 2001] also provides an expression to calculate the losses due to the motor seals as:

$$P_s = 52{,}33\mu F n d_s \times 10^{-3}\ W \qquad \text{Eq.(3-62)}$$

where μ is the coefficient of friction, F is the force between the seal and the end-shield, n is the rotational speed of the motor, and d_s is the average seal diameter.

Windage losses result from the aerodynamic effects caused by salient rotor geometry churning the surrounding fluid (air) as it spins. If a cooling fan is used, it also adds to these losses marginally. Evidently, windage losses are highly dependent on the operating speed. Although many researchers have attempted to provide analytical formulations to calculate these losses, the most reliable way to estimate these losses is via experiments. [Raulin, et. al., 2004] provide a general expression for friction and windage loss as:

$$P_{fw} = a_w V_{air}\omega^q \qquad \text{Eq.(3-63)}$$

where a_w is a windage coefficient, V_{air} is the volume of air in the motor assuming that the stator is full, ω is the motor operating speed and q is a curve-fitting parameter.

[Gieras, 1999] provides an approximation to calculate the windage loss (by assuming a cylindrical rotor) as:

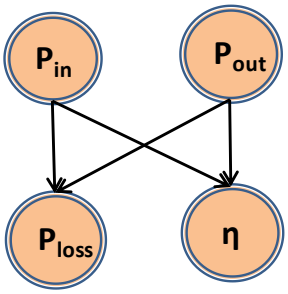
$$P_{wind} = \pi C \rho \Omega^3 \left(\frac{D_{rotor}^2}{2}\right) w \qquad \text{Eq.(3-64)}$$

where C is the friction coefficient, ρ is density of the cooling gas, $\Omega = 2\pi n$ is the angular velocity of the rotor, D_{rotor} is the rotor outer diameter and w is the axial length of the rotor. The friction coefficient is influenced by the turbulence in the air gap flow as well as the surface quality of the rotor and stator. [Gieras, 1999] states that for surface speeds < 200 m/s, the coefficient can be estimated as:

$$C = \frac{0.0152}{R_e^{0.24}} \text{ , where } R_e = \frac{\rho \Omega D_{rotor} g}{2\mu} \qquad \text{Eq.(3-65)}$$

where R_e is the Reynold's number , μ is the dynamic viscosity of the cooling gas (air or helium), and g is the air gap length.

The contribution of each of these to the total losses is different at different motor operating speeds. Typically, the electrical losses dominate at lower to moderate speeds (copper losses tend to dominate at the lower end of the range and core losses at higher speeds) whereas at very high speeds, mechanical losses dominate. Other operating factors also influence the composition of losses. [Materu and Krishnan, 1992] provide a comprehensive analysis for an estimation of SRM losses under all conditions including using effects of stator current waveforms, conduction angle, speed, saturation, etc. [Hasegawa et. al, 2002] demonstrate that optimizing the applied voltage decreases copper losses but results in an increased value of iron losses. By judiciously varying the various control parameters (e.g. based on performance maps as described in [Tesar, et. al. 2005]) for the SRM like the phase turn-on and turn-off angles, the supply voltage, the duty cycle and frequency of the applied



P_{in} = Input Electrical Power

P_{out} = Output Mechanical Power

P_{loss} = Total Power Losses

η = Motor Efficiency

Figure 3-21: Relation between Input Power, Output Power, Losses and Efficiency

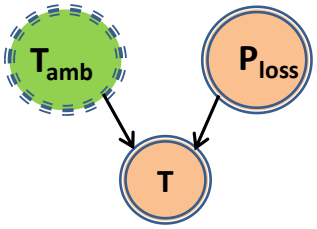
PWM signal, etc., based on the prevalent operating conditions, it is possible to minimize these different losses. Depending on the sensors used in an application, the efficiency of the SRM can be estimated in realtime by determining each of the different types of losses described above. Alternatively, with just the knowledge of the input electrical power and the output mechanical (shaft) power, the overall efficiency of the SRM can be calculated as:

$$\eta = \frac{P_{out}}{P_{in}} = \frac{P_{in} - P_{loss}}{P_{in}} = \frac{P_{out}}{P_{out} + P_{loss}} = \frac{P_{mech}}{P_{elec}} \qquad \text{Eq.(3-66)}$$

where P_{loss} represents the sum of all the losses described above (Figure 3-21).

3.3.8. Relation between Power Loss and Motor Temperature

Usually the focus is on the electromagnetic and mechanical aspects of the motor during the design phase as well as during operation. Due to its complex and nonlinear nature, thermal analysis for such systems is often not given due importance. The different types of losses that occur during the conversion of electrical input energy to mechanical output power as described earlier are primarily dissipated in the form of heat, which leads to an increase in the motor temperature. The heat generated is typically not concentrated and is unevenly distributed across the different motor components like the bearings, the windings, the stator iron, the rotor etc. The knowledge of this distribution can help provide guidance on the cooling method to be used, the values of current, frequency that can be used for operation, etc [Miller, 1993]. In order to determine the temperature rise of the different motor components, the common approach is to create



T_{amb} = Ambient Temperature

P_{loss}= Total Power Loss

T= Average Motor Temperature

Figure 3-22: Relation between Power Losses and Motor Temperature

a thermal equivalent circuit for analysis using lumped parameters [Miller, 1993][Rouhani, et. al, 2007] or via FEA [Wu, et. al., 2003][Inamura et. al., 2003].

An increase in the motor temperature not only affects the motor efficiency adversely but may also result in an overall degradation in the motor performance. [Balamurugan et. al, 2004] state that the influence of increased temperature on the electromagnetic characteristics of a SRM is not significant under light operating conditions (because of the time constants of thermal phenomena are orders of magnitude different from those of electrical or magnetic phenomena) but may be crucial when the motor is operating under high loads since the temperature rise under such conditions may be significant enough to influence these characteristics. Although there is no danger of demagnetization of magnets at high temperatures as in the case of brushless DC motors, high temperatures are still a concern in SRMs since it can affect the winding insulation, the bearing lubrication characteristics, and could potentially cause a complete motor failure.

For a constant current, the copper losses can increase significantly with an increase in the temperature since the winding resistance increases [Miller, 1993]. [Lunifeld and Tesar, 1995] show that motor temperatures can vary from 25^{o}C to greater than 150^{o}C in EMAs under different operating conditions. With such drastic changes in temperature, the increase in resistance may be as much as 20% for a 50^{o} C rise (and 53% for a 135^{o}C rise) in temperature in case of copper windings, causing a corresponding increase in the copper losses, thus setting up a vicious cycle [Miller, 1993]. [Brancato, 1992] states that the thermal life of an insulation system (rated for a given temperature) is halved for every 10^{o}C increase in temperature. Typically motors use grease lubricated bearings instead of oil lubrication, where the oil is fed by an active lubrication system. Increased temperatures may reduce the lubricant viscosity in bearings making it more difficult for a lubricant film to be formed on the rolling surfaces of the bearing, which in turn leads to higher friction and a further rise in temperatures. Higher bearing temperatures may lead to a reduction in the operational clearance in the bearings due to thermal expansion and may ultimately result in bearing seizure. Analysis by [Kang and

Tesar, 2004] shows that the bearing load vs. deflection characteristics are also significantly influenced by temperature changes.

The magnitude of rise in temperature depends on how effectively the heat generated in the motor is removed. With effective heat dissipation, issues like variation in resistance can be addressed, which in turn allows a higher current carrying capacity for more torque production. Since the SRM has active power coils only on the stator, most of the electrical losses are localized to the stator and its windings; the rotor remains relatively unaffected and runs cooler on average. Heat transfer is achieved by a mix of conduction (from the stator coils to the motor shell, from the shell to the mounting frame, from the shaft to the bearings to the motor shell, etc.), convection (from the motor shell to the surroundings, or to the cooling fluid, if active cooling is used[16]) and radiation (from the motor shell to the surroundings).

The amount of heat dissipated by conduction can be approximated as [Miller, 1993]:

$$Q = kA\frac{dT}{dx} \approx kA\frac{\Delta T}{t} \qquad \text{Eq.(3-67)}$$

where ΔT is the temperature difference assuming that the heat is being conducted across a the block of thickness t and area A, k is the thermal conductivity which depends on the material and is also a function of the temperature. In case of the SRM, conductive heat transfer occurs from the conductors to the stator core and further to the motor shell. The temperature distribution in each of these components and at the interfaces between them can be determined using equivalent thermal resistances for each of them [Miller, 1993]. The heat generated in the windings has to pass through several layers before it reaches the motor shell. The windings are often coated with a resin of high thermal conductivity to alleviate this problem to a certain extent.

From the motor shell, the heat is transferred to the surroundings via convection is given by the relation:

[16] In applications where the EMAs operate under significant duty cycles, to achieve higher power/ torque-densities, active cooling using oil or other liquid coolants is often employed

$$Q = hA\Delta T \qquad \text{Eq.(3-68)}$$

where $\Delta T = T - T_{amb}$, is the temperature difference between the cooling medium (air, in most cases) and the surface being cooled, and h is the convective heat transfer coefficient. Its value is determined by the viscosity, thermal conductivity, specific heat, and velocity of movement of the coolant [Miller, 1993]. This is termed as free convection where the flow of the coolant is not aided by pumps, fans, etc. as compared to forced convection where one of these is used. [Miller, 1993] estimates h for free convection around a horizontally mounted, unfinned cylindrical motor with diameter D as:

$$h \approx 2.14 \times 10^{-3} \left(\Delta T / D\right)^{1/4} \qquad \text{Eq.(3-69)}$$

With forced convection of air, the value of h increases almost six times, based on the air velocity. If V is the air velocity and L is the frame length of the motor, then the corresponding h is given by [Miller, 1993] as:

$$h \approx 11.2 \times 10^{-4} \sqrt{V / L} \qquad \text{Eq.(3-70)}$$

The total heat transferred by radiation is given by the Stefan-Boltzmann equation as:

$$Q = Ae\sigma(T^4 - T_0^4) \qquad \text{Eq.(3-71)}$$

where σ is the Stefan-Boltzmann constant for a black body, e is the coefficient of emissivity (a measure of the effectiveness of radiation from a body relative to that of a perfect black body), T is the absolute temperature of the heat radiating surface and T_0 is the absolute temperature of the surroundings. The amount of heat transfer via radiation is typically lesser than the heat transferred via convection.

As seen from the equations for each mode of heat transfer, the total heat dissipated from the interior of the motor to its shell and eventually to its surroundings depends on the surface area of the motor shell. Fins are sometimes used to increase this area without drastically affecting the overall size of the motor. The other critical factor that determines the total heat transferred from the motor is the temperature of the ambient medium. Thus, for a given motor geometry and cooling arrangement, the average temperature at which the motor operates is primarily determined by the overall magnitude

of motor power losses which dictate the amount of heat generated and the ambient conditions. This relation is shown in Figure 3-22.

3.4. CHAPTER SUMMARY

The development of a comprehensive Bayesian network model for a four phase switched reluctance motor was presented in this chapter. Diverse facets like torque generation, acoustic noise generation, motor dynamics, and thermal considerations were examined in detail using mathematical models available in literature to derive the corresponding causal representations for each of these aspects individually. The smaller causal networks were then combined to generate the complete network representing the test set-up in its entirety (Section 3.2). The different models discussed in this chapter and the relevant sections are shown in Table 3-1.

Since modeling the system was the prime objective in the present case, a causal approach was adopted to make the resultant network intuitive to the end-user as well as represent the actual interaction between the variables in the physical system as closely as possible. However, a number of other criteria driven by factors like application requirements, need for fail-safe operation, availability of specific types of performance maps, etc., may influence both the structure of the network itself as well as how well it can be utilized in an application. A few of these criteria will now be explored in further detail in Chapter 4.

Table 3-1: Chapter Summary

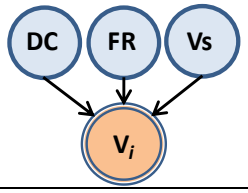
Sub-network	Notation	Description	Section
	i = 1…n = Number of motor phases, DC = PWM Duty Cycle, FR = PWM Frequency, V_s = Supply Voltage	Influence of Control Parameters on Motor Phase Voltage	3.3.1

Table 3-1: Chapter Summary (continued)

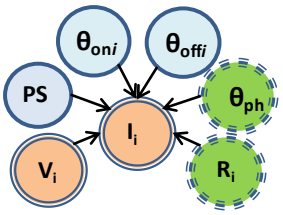
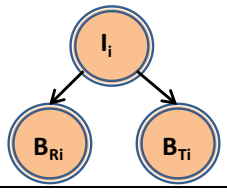
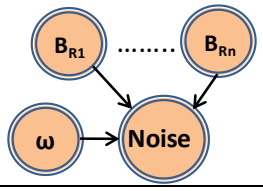
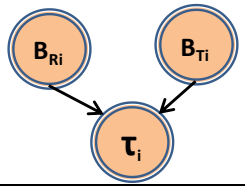
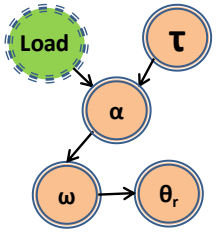
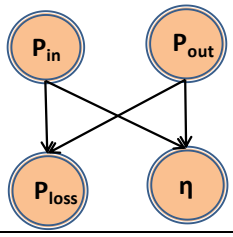
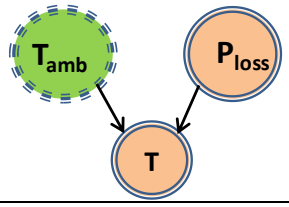
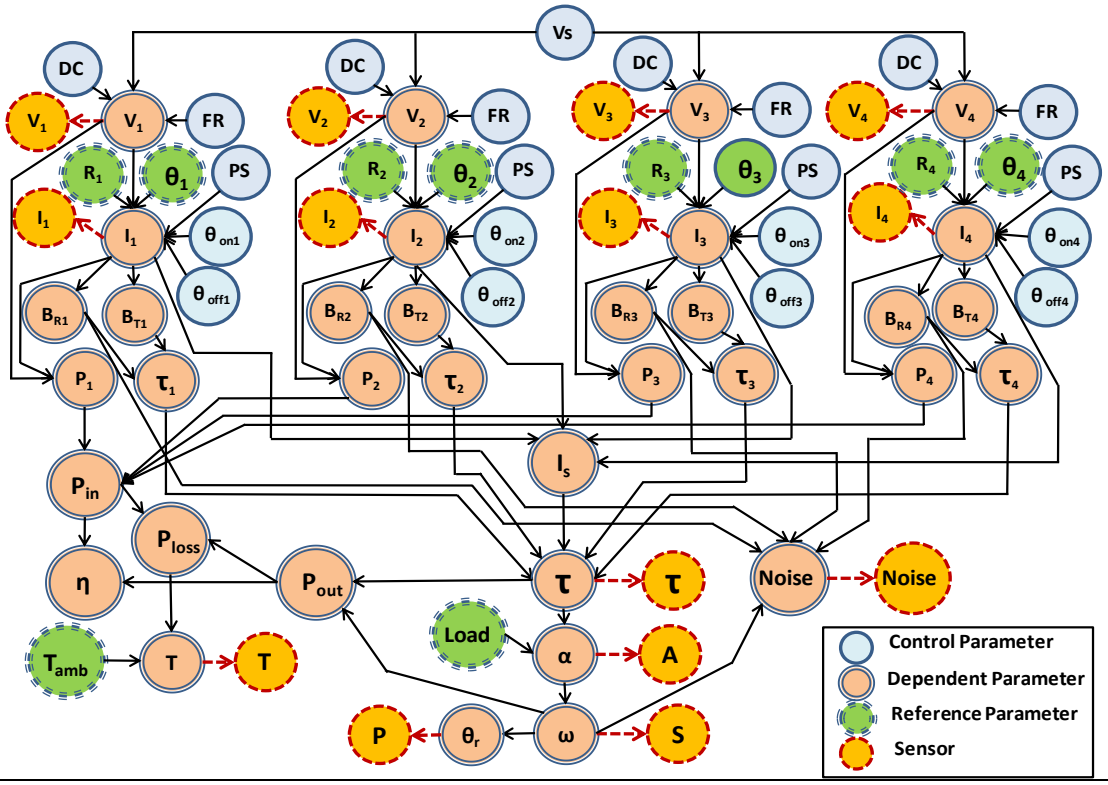
Sub-network	Notation	Description	Section
	i = 1…n = Number of motor phases, V_i = Phase Voltage, R_i = Phase Resistance, Θ_{ph} = Rotor Angle Relative to Stator, PS = Phase Status (On/Off) Θ_{offi} = Phase Turn-off Angle Θ_{oni} = Phase Turn-on Angle I_i = Phase Current	Relation between Phase Voltage, Phase Current and Phase resistance	3.3.2
	i = 1….n= Number of motor phases, I_i = Phase Current B_R = Magnetic Flux Density, radial B_T = Magnetic Flux Density, tangential	Relation between Current and Flux densities	3.3.3
	i = 1….n = Number of motor phases, B_R = Magnetic Flux Density, radial ω = Motor Speed, N= Acoustic Noise	Relation between Flux Density, Speed and Acoustic Noise	3.3.4
	i = 1….n = Number of motor phases, B_R = Magnetic flux density, radial B_T = Magnetic flux density, tangential τ_i = Torque produced by a single phase	Relation between Flux Density and Torque	3.3.5
	τ = Total Electromagnetic Motor Torque α = Angular Acceleration of Rotor ω = Angular Velocity of Rotor θ_r = Rotor Position $Load$ = External Load Torque	Relation between Motor Torque, Acceleration, Speed and Position	3.3.6
	P_{in} = Input Electrical Power P_{out} = Output Mechanical Power P_{loss} = Total Power Losses η = Motor Efficiency	Relation between Input Power, Output Power, Losses and Efficiency	3.3.7

Table 3-1: Chapter Summary (continued)

	T_{amb} = Ambient Temperature P_{loss} = Total Power Loss T = Average Motor Temperature	Relation between Power Losses and Motor Temperature	3.3.8



Detailed Bayesian Network Model of a Four Phase Switched Reluctance Motor Obtained by Combining the Smaller Sub-models (Section 3.2)

Chapter 4. Criteria Based Resource Management

4.1. INTRODUCTION

The development of a Bayesian network for a switched reluctance motor was presented in Chapter 3. The resultant network structure (causal) was dictated purely by the motor physics. However, for a network to be of practical use in many applications, there may be other factors that not only influence its configuration in terms of the number of nodes and the interconnections between them, but also the manner in which the parameters of the different nodes are populated, the choice of appropriate inferencing algorithms to use for the available network structure, etc. This chapter explores some such issues that may need to be factored in, when designing and utilizing a Bayesian network for representing and making operational decisions for any system of interest. Section 4.2 defines the problem of resource management. This is followed by Sections 4.3 and 4.4 which provide descriptions of the different criteria that can be used to achieve this objective from design and operational perspectives respectively.

4.2. RESOURCE MANAGEMENT

Given the numerous parameters associated with even a moderately complex system, a full suite of sensors is often required to enable comprehensive monitoring of the system. There are three main facets to consider in the use of multiple sensors as outlined in [Xiong and Svennson, 2002][1]:

- *Sensor deployment*: To address when, where, and how many sensing resources are needed to respond to changes in the monitored system and/or its operating environment. This typically happens during the system design phase
- *Sensor assignment*: Given finite sensing resources, plan sensor functions and adapt sensor usage according to changing mission requirements. This may involve decisions on the best way to correlate the information collected from the different sensors during the system certification phase

[1] The focus in the authors' work was sensor fusion in a target tracking application

- *Sensor coordination*: Sharing information between sensors to make the best use of the available information when the system is in operation.

These aspects are driven by the combined (and often conflicting) objectives of effectively addressing a system's feedback needs, keeping the costs and complexity to a minimum and without any compromises on the overall reliability or performance of the system. There is thus a need for a criteria-based resource management framework that can help decide upon and best utilize the available sensors and other resources at any given time.

The term 'sensor management' is often used in the context of wireless sensor networks to refer to the process of scheduling and activating the appropriate sensors within a group of sensors distributed over a wide geographical area to address issues like energy consumption, limited bandwidth, etc. or in the context of target tracking where it refers to the process of selecting appropriate sensors, modalities, etc. to optimize their effectiveness in characterizing the probability of a target occurring in a region under consideration [Onel et. al, 2009], [Kolba and Collins, 2006], [Vaidya, et. al. 2005] [Kreucher, et. al. 2003]. The common thread in the examples above is that application specific criteria are used to make decisions on what sensors to use, when and for which purpose. Many of these applications use criteria/norms derived from field of information theory (such as entropy, mutual information, Kullback Leiber or Renyi divergence, etc.) in combination with some form of estimation theory (Kalman filter, particle filters, etc.) [Augenbaugh and La Cour, 2008]. In addition, researchers have also explored alternative approaches to sensor management in other domains such as the use of geometric interaction between sensors and the environment in conjunction with Bayes reasoning for sensor selection in a robotic system [Giraud and Jouvencel, 1995], the use of a gating neural network and a rules/knowledge database to estimate the reliability of sensor readings and the sensors to be used in a surface grinding process [Kobayashi, et. al. 2002], using an empirical Bayes procedure for fault detection in diesel engines [Subrahmanya, et.al., 2010], using a decision-theoretic approach based on user defined criteria for surveillance [Spaan and Lima, 2009], using soft computing/fuzzy logic techniques in aircraft sensor management [Oosterom, et. al, 2002], etc.

In the present context, the objective of resource management is to enable the selection and coordination of a suite of sensors to monitor a dynamic system using well-defined criteria during both the design phase as well as during the operational phase. With such an approach, it is possible to provide a synergy (via functional redundancy) that can enable the use of different sensors to corroborate each other as well as influence the availability of information in situations where data from specific sensors becomes partially or completely unavailable due to sensor or connector failures, bandwidth/power constraints, etc. Criteria based resource management can help in the selection and use of a finite set of sensors in concert with available computational resources to maximize the utility of the information available to the operator regarding the system at any given time (by providing guidelines that the system designer or operator can use to make decisions such as determining the priority of a sensor or a subset of sensors to address a specific objective). It can also help avoid overwhelming data transfer, computational and memory requirements in a multi-sensor system by ensuring that only the truly essential data is acquired and utilized.

4.3. DESIGN CRITERIA

[Korb and Nicholson, 2004] describe the process of construction of a Bayesian network for any system, first put forth in [Pearl, 1988], as an algorithm consisting of the general steps as shown below:

Pearl's Bayesian Network Construction Algorithm

1. Determine the set of n variables $\{S_i\}$, where i =1,2,….n, that are relevant to the domain under consideration
2. Determine the ordering of the variables $<S_1, S_2, S_3 \ldots S_n>$
3. Loop through 1 to n
    a. Insert S_i into the network
    b. Create arcs from the minimal set of nodes that are already present in the network $\Pi(S_i)$ such that the variable S_i is conditionally independent of all its non descendants given its parents $\Pi(S_i)$ i.e.

$P(S_i| S_1, S_2, S_3 \ldots S_{i-1})$=P($S_i$| Π ($S_i$)) where Π ($S_i$) ⊆{ S_1, S_2, $S_3 \ldots S_{i-1}$}

c. Define the conditional probability table for S_i

End loop

For most engineering applications, the nodes in the Bayesian network represent the different physical parameters of interest, for which sensors are integrated into the system. A network composed of these measurands/variables therefore needs to be designed concurrently with the design of the actual system itself. It needs to mirror the actual system as closely as possible, since it is meant to represent the behavior of the system for decision making during operation. The process thus tends to be iterative, as there are numerous criteria that need to be balanced simultaneously[2].

The following section discusses some design criteria that may be used not only to determine the choice of sensors while designing the physical system but also to address some of the requirements to create a Bayesian network representation for it as per Pearl's network construction algorithm i.e. determining relevant nodes, their ordering, directing the links appropriately, defining the node parameters, etc. (Note that these criteria are only a representative list to provide some guidelines that may be used to create and refine the network. Numerous criteria have been proposed and applied successfully in many applications, for instance, in [Jensen and Nielsen, 2007][Rajabally, et.al, 2004][Neil, et. al., 2000] and others. More application-specific criteria may be defined in the future)

4.3.1. Application Requirements

The number of sensors in any system is principally dictated by the feedback requirements of the application itself. In some applications, it is imperative to include certain sensors for effective system operation. These may be termed as 'essential' sensors, for without them system operation may not be possible. For instance, for conventional speed control of a brushless DC motor, the use of a position sensor (to track the rotor position for commutation) in conjunction with voltage sensors for each motor phase (to track the voltage signal applied across each phase which in turn dictates the

[2] In the present case, it is assumed that the network is created by experts with in-depth domain knowledge.

overall motor speed) is vital. In case of robotic manipulators which are used for precision operations like positioning or assembly, accurate joint position sensors or force-torque sensors are a must. In addition, there may be other sensors that can be integrated into the system for more comprehensive monitoring purposes. For example, temperature sensors may be integrated into motor windings to monitor power lost in the form of heat (which is detrimental to the motor windings) or an acoustic noise sensor to determine the increase in the noise generated by the motor as the operating speed increases (due to additional windage effects). These can be referred to as the 'optional' sensors. Appendix A shows the typical set of sensors recommended for a general class of electromechanical actuators consisting of the four major components as indicated [Tesar, 2009]. Depending on the components that are more critical than others for a specific application (and hence need to be controlled or monitored exhaustively), the set of essential and optional sensors can be chosen from this array of possible choices.

Also included in the category of optional sensors are sensors that may be used to indirectly estimate parameters for which physical sensors do not exist or are difficult or expensive to integrate into the application[3]. For example, if motor efficiency is an important criterion in an application, it may be estimated using current and voltage sensors to determine the power input to the system and torque and speed sensors to compute the output power from the system. In some applications, especially in those where human safety is involved, hardware failures cannot be tolerated (e.g. space applications) and it is often imperative to have hardware redundancy for all the sensors in the system. To enable condition based maintenance of systems using performance maps and envelopes, it is necessary to have sensors corresponding to the measurands represented in those maps, so that they can be continuously updated to reflect the extant condition of the system at any given time. The number of all such essential and optional

[3] The common approach in most application is to use off-the-shelf commercially available sensor options wherever possible to keep the overall system development costs low (during the design phase). Pursuing special purpose sensor designs is usually only done in one-off cases where there are no other alternatives

sensors, taken together determines the total number of nodes in the network[4]. The size of the network has a direct bearing many factors that influence the operational use of the network like memory/storage requirements (for the different CPTs), choice of inferencing algorithms, computational requirements, ease of understanding or intuitiveness to the operator, etc.

4.3.2. System Design Constraints

In any application, it is desirable to incorporate as many sensors as possible (without impeding system operation) to provide the most comprehensive monitoring of the system's various operational parameters. This may not be difficult for a laboratory test setup but in many cases, the inclusion of additional sensors other than the most essential ones is often restricted by the size (overall volume) and weight constraints imposed by the application. For example, in case of EMAs used in aircraft or small diameter (0.25" to 2") EMAs used in precision surgical robots [Tesar, 2010], etc. both weight and volume are at a premium. There is also the need to account for any design modifications that may be required to accommodate the additional cabling required for power supply and signal transmission requirements (or analogously the additional on-board power and transmission required for wireless sensors), etc. all of which typically contribute towards increasing the overall dimensions and weight of the system, for example as described in [Iaconis and Tesar, 1991], [Nowak and Tesar, 1998], etc.

The inclusion of more sensors usually calls for a number of compromises to be made during the system design process to accommodate the sensors as effectively as possible without compromising the system performance. This is often encountered when the sensors have to be close to or located at the stimulus (for example, temperature sensors like thermistors or magnetic field sensors to measure the field strength in a motor air gap). In such cases, careful design is required to integrate them into the system as

[4] It is assumed that every node is 'measurable' directly by a sensor or can be determined analytically using the available measurands, in order to be able to update the corresponding node parameters and performance maps. For nodes which do not have a direct physical sensor associated with them, this includes the case where the human expert's opinion is taken as the definitive value for the variable represented by that node.

compactly as possible while simultaneously protecting the sensors as well as the on-board electronic circuits from potential damage due to factors like excessive vibrations, proximity to components that dissipate heat, etc. In some cases, although a particular variable may be considered crucial to the overall system operation, the available off-the-shelf sensors may not be suitable for compact integration into the system or they may be too fragile in their construction (which may be an issue when deployed under real-world conditions). This may call for the development of a completely new sensor design to satisfy the dimensional constraints imposed by the application. A pertinent example of this situation is the limited variety of rotary torque sensors in suitable form factors that can be compactly integrated into modular EMAs [Tesar, 2010].

In such scenarios, choices often have to be made a) whether to leave out some sensors from the system completely, b) if it is worthwhile to invest in a new sensor design, c) if some of the existing design constraints may be relaxed to accommodate available sensors, d) if the variables of interest can be analytically inferred to an acceptable degree by integrating other sensors which can still satisfy the existing design constraints (for instance, inferring the EMA output torque using the current drawn by the electric prime mover in it, after factoring in the gear ratio), etc.

4.3.3. Overall Costs

Although the cost of sensors (as a fraction of the overall system cost) is much lesser than that of other parts of a system in general, the outlay for some critical sensors may be a significant portion of the overall system budget (for instance, high accuracy position sensors or rotary torque sensors for use in precision actuators are typically more expensive other components). Integrating additional sensors, other than the essential sensors, typically implies an increase in the overall system cost, since it not only includes the actual cost of the sensors but also the auxiliary equipment needed i.e. power supply sources, additional cabling/channels needed for data acquisition from all the sensors, computational resources needed to handle the extra data, etc. As mentioned earlier, special provisions may need to be made to the overall system design to accommodate

more sensors which in turn may also add to the total cost. The overall cost may not be a significant factor in special purpose or one-off system designs, but for systems which are more widely deployed it can often be an overriding factor. This may necessitate the exclusion of certain sensors from being integrated during the design phase.

In such cases, it may still be possible to include the nodes corresponding to the absent sensors into the Bayesian network during the model design phase if a definitive source of information for that variable is available (in the form of expert opinion, performance maps or accurate analytical relations from which they can be inferred using other variables)[5]. Another approach that is often adopted is to use lower cost alternatives for the more expensive sensors. Such sensors may not provide the same high standards of performance but may still be used to provide useful information which may be of lower 'quality' (resolution, accuracy, etc.) and considered acceptable for the application. A relevant example is the use of lower cost hall-effect sensors as compared to precision position encoders for motor commutation.

4.3.4. Relative Importance of Sensors

As discussed earlier, in any application, there are essential sensors without which it may be impossible to achieve satisfactory system operation and additionally there may be optional sensors that are used to monitor some secondary parameters of interest to enable enhanced system performance. For instance, Table 4-1 shows the importance of different types of sensors for a general class of intelligent EMAs (rotary), considering their use in a variety of domains as explored in [Tesar, 2009]. It can be seen that the position and torque sensors are the two most critical sensors in such systems.

In certain applications, the sensors corresponding to the critical variables of interest may be too fragile and may be prone to frequent failure or loss of performance (for instance, high precision position encoders are usually sensitive to high operating temperatures). Any degradation or unexpected loss of information from such a sensor

[5] However, caution must be exercised if such an approach is adopted because it may not be always possible to accurately update the node parameters or maps when the system is in operation, due to the lack of sufficient information

vital to the system, may lead to undesirable system behavior or in the extreme case, a catastrophic system failure.

Table 4-1: Relative Importance of Sensors in Intelligent Electromechanical Actuators

Sensor	Significance to Intelligent EMAs	
Position	10	Note: Scale of 1 to 10, with 1 being the lowest
Torque	9	
Current	8	
Voltage	7	
Velocity	6	
Temperature	5	
Acceleration	5	
Noise	4	
Vibration	4	
Magnetic Field	3	

In such situations, if the sensors are too expensive to replace or are located in an inaccessible location within the system and it is not possible to replace or repair them when the system is in operation without other consequences (altering the system, downtime costs incurred as a result of shutting down the system for repair, etc.), it is desirable to provide some failsafe provision for obtaining these critical measurands, in case of a loss of information from their corresponding sensors.

(a)

(b)

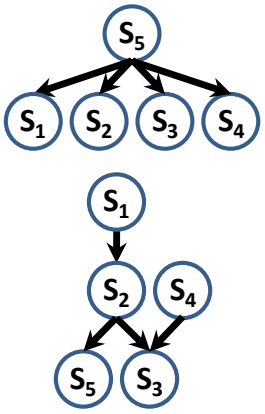


Figure 4-1: Fault Tolerance for Critical Sensors

With the use of a Bayesian network to provide functional redundancy, data from one or more of the other operational sensors can be used to set evidence to the network and the value of the node corresponding to the sensor of interest, say S_I, can be determined using probabilistic inferencing as described in Chapter 2. In terms of the network structure, this means that the node S_I must be related to as many other nodes as possible. The objective is to provide as many alternative sources of information as possible to infer

the critical measurands so that failsafe operation is possible[6]. Different network structures can produce data of differing quality. The most suitable network would be one where the value of S_1 can be obtained from the node/s which can be potentially set as evidence, without the need to traverse through a lot of intermediate nodes or links[7]. As an example, consider two possible network structures representing relation between five variables of interest S_1, S_2...S_5, (Figure 4-1) with S_5 being the most critical measurand. Consider the case where there is a loss of information from the sensor corresponding to S_5. From Figure 4-1 (a) it can be seen that in, the value of the node S_5 can be inferred using data from any of the sensors corresponding to nodes S_1 through S_4 with only one intermediate link involved. The uncertainty in the inferred value of S_5 is determined by the relationships S_5->S_i as encoded in the conditional probabilities $P(S_i|S_5)$, where i =1, 2, 3, 4. Even if one or more of the other sensors S_1…S_4 become partially or completely unavailable, an alternative always exists to infer the value of S_5 (except in the extreme case where all the sensors S_1…S_4 become unavailable). However, in Figure 4-1 (b), the best option available to infer the value of S_5 with least uncertainty is by setting the value of the sensor corresponding to S_2 as evidence to the network. Although any of the other sensors S_1…S_4 may still be used to infer the value of S_5, if the sensor corresponding to S_2 also becomes unavailable, the uncertainty in the inferred value will be higher.

4.3.5. Type of Data Available

As seen earlier, the degree to which functional redundancy can be achieved for different variables in the Bayesian network for any system is determined to a large extent by manner in which the nodes are connected with each other. These interconnections are, in turn, often dictated by the form in which data to populate the parameters of the

---

[6] Although any node in the network can be set as evidence to infer the value of S_1, the choice of which sensors may be used to set evidence may be driven by other factors. For instance, it may not be advisable to use a node whose corresponding sensor has a low accuracy, as evidence. The inaccurate sensor may result in an incorrect node state being set as input to the network, resulting in an erroneous inferred value of S_1.

[7] The resultant value obtained for S_1 and its uncertainty as given by the spread of its probability distribution, is directly proportional to the number of intermediate links between S_1 and the evidence node/s as well as the conditional probability distributions of all the intermediate nodes. These factors are investigated in greater detail in Section 4.4

different nodes in the network is available (or the manner in which it can be acquired), and based on the judgment of the domain expert who creates the network. In the present context, the construction of links between the different nodes in the network depends on how the available data has been structured in the form of different performance maps[8] since the probabilistic models (conditional probability values) that are encoded in the CPTs at each node are derived from the same data used to generate these maps.

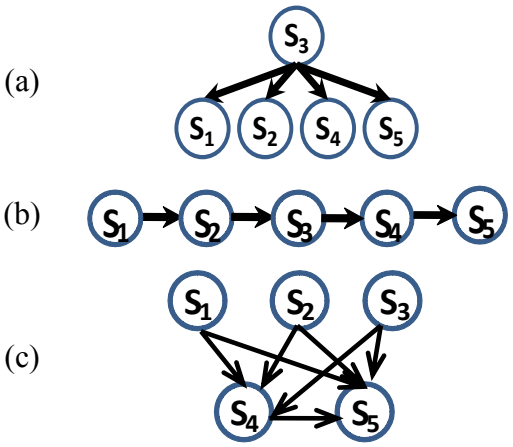


Figure 4-2: Different Network Structures Based on Available Data

Consider a system with five variables of interest S_1, S_2…S_5. Figure 4-2 shows a few of the many possible configurations in which the nodes representing these variables can be linked[9]. Suppose the sensor corresponding to a variable, say S_5, suddenly becomes unavailable (say, due to a sensor failure) and the value of S_5 has to be inferred from the sensor readings corresponding to one or more of the remaining variables. In this case, the performance maps/data relating S_5 to all possible combinations of those variables must be available. For instance, suppose, based on the domain expert's opinion, the value of the variable S_5 can be best estimated using the value of the variable S_3. With a structure as shown in Figure 4-2 (a), to infer the value of S_5 from S_3, only data relating S_3 to S_5 as represented by the S_3->S_5 link is needed (i.e. $P(S_5|S_3)$ ∀ S_5 and S_3). In Figure 4-2 (b), to infer S_5 from S_3, data representing two relations, S_3->S_4 and S_4->S_5 i.e. $P(S_4|S_3)$ ∀ S_4, $S3$, and $P(S_5|S_4)$ ∀ S_5, S_4) are needed. In

[8] Specifically, the manner in which the data has been grouped in different combinations of control, reference and dependent parameters as described in [Ashok and Tesar, 2007]

[9] Here the links between nodes only serve to indicate the dependence between the variables as determined by the domain expert based on the available data and not necessarily a cause-effect type of relation. Many structure learning algorithms exist in literature to determine the network structure from available data and depending on the different metrics to evaluate the resultant intermediate results, these algorithms can result in vastly different network structures derived from the same set of data [Acid, et. al, 2004].

Figure 4-2 (c), in addition to the data relating S_3->S_5, the data representing the relations S_1->S_5, S_2->S_5, and S_4->S_5 are also needed simultaneously (i.e. $P(S_5|S_1, S_2, S_3, S_4) \forall S_5, S_1, S_2, S_3$ and S_4) to produce the best estimate of the value of S_5 given the available data.

The availability of these different relations is determined by the number as well as the type of experiments that need to be conducted in order to collect the requisite data. Ideally, if the data corresponding to all the nodes is acquired synchronously during system certification (when the 'as-built' performance maps are generated), the relation between any pair of variables/nodes in the network (i.e. the CPTs containing $P(S_i|S_j)$ where $i \neq j$, and i, j=1,2…5) can be developed relatively easily. However, in certain situations this may not be possible. Since the process of experimentation is time and resource intensive, it may be practical only to perform a few experiments to measure only the most critical parameters needed for system certification [Janardhan and Tesar, 2008]. There may also be cases where all the sensors used during the testing process may not be available when the system is deployed outside a test setup thereby rendering some of the generated maps unusable (because they cannot be accurately updated). In some cases, the testing protocol or the hardware used during system certification may be such that the variables of interest can be measured only one at a time or in specific groups based on the changes in certain control parameters (for example, as described in [Janardhan and Tesar, 2008] in generating maps for a brushless DC motor), etc. In the extreme case where neither empirical data nor accurate analytical relations are accessible relating certain combinations of variables, then such variables cannot be linked to each other in the network. Thus, the constitution of available data can influence the network structure.

4.3.6. Causality

The topology of a Bayesian network is highly influenced by the ordering of the nodes that represent the variables in the system under consideration. Strictly speaking, the links in a Bayesian network only represent the conditional independencies between the connected nodes and need not necessarily represent causal relationships between those nodes. However, using causal relations to represent the link between the nodes can help

attribute physical meaning to the values that are obtained using the network, making it more intuitive for the user to comprehend those values and use them in decision making.

For instance, consider a network with two nodes, current and torque, representing a motor. Assume that comprehensive experimental data regarding both the variables is available over the entire operating range in an application where the motor is used and can be used to create the required CPTs. The relation between them can be represented as two possible network structures as shown in Figure 4-3 (a) and (b). From a mathematical perspective, both the above networks are equally valid since both forward and inverse probabilistic reasoning based on available information i.e. $P(Torque|Current)$ or $P(Current|Torque)$) are possible by simply using the CPT or Bayes' theorem, as the case may be. But for both experts (who are involved in designing the system and its Bayesian network representation) and non-experts (who may be the end users making the final decisions for operating the system), the structure shown in Figure 4-3 (a) will provide a greater intuition in decision making since it represents what actually happens in a motor i.e. the current applied across the motor windings results in torque generated by the motor (due to the air-gap magnetic field) and not the other way around; with the torque generated being directly proportional to the magnitude of the supplied current.

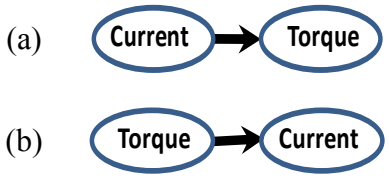


Figure 4-3: Causality

Various authors including [Pearl, 1988], [Korb and Nicholson, 2004], [Neapolitan, 2003], etc., have highlighted the opinion that more often than not, a causal model underlies any real-world joint probability distribution and typically results in a Bayesian network that can be considered practically useful. [Pearl, 1988] emphasizes that use of such causal schema minimizes the number of relationships that need to be considered to model any system, thus resulting in compact networks (i.e. one displaying more independencies than a non-causal representation) with lesser interlinking between the nodes (no unnecessary or redundant links). The simpler structure in turn, has a significant influence on the inferencing process (more compact CPTs, faster calculations

in general). [Kenny, 2004] outlines the three conditions which may be used to determine whether a variable *A* causes another variable *B* or not (and hence also examine the direction of the link between *A* and *B* in a Bayesian network):

i. *Precedence in time*: For a variable *A* to cause a change in a variable *B*, *A* must temporally happen before *B*. This implies that the causal relation is asymmetric.

ii. *Functional relationship*: There must be a function relationship between the cause and the effect parameters ($B=f(A)$). If the knowledge of one variable does not provide any additional information regarding the other variable, then they can be considered as independent of each other. If not, then they are related.

iii. *Non-spuriousness*: The relation between *A* and *B* should not be influenced by the presence of a third variable *C* that causes both *A* and *B*, such that if C is controlled, then *A* and *B* become independent[10]

4.3.7. Discretization of Data

An important factor that determines the quality of information inferred using the Bayesian network is the discretization of the data encoded within each node in the network. The aspects to be considered in discretizing a random variable corresponding to a node include the number of states (discrete levels) or bins that the available data has to be divided into as well as the limits/sizes of the individual bins. The number of states is typically determined by the range of interest for each variable in the application. The nodes in a network usually have a differing number of states, since physical variables tend to vary at different scales. Evidently, the greater the number of bins, the more detailed is the information regarding the variables of interest and greater is the resolution of values that can be inferred using the network. The limits of the bins for each node also need to be relevant in terms of the application being considered. These limits represent the acceptable range of deviation from the steady state value for each state of every node. It is evident that with a smaller level of discretization (i.e. discretizing into a large number of bins/states), the size of the CPTs that need to be stored and manipulated each

[10] Note that this is different from intervention, where *A* causes *C* and in turn, *C* causes *B*. [Kenny,2004] states that spuriousness explains away causality whereas intervention elaborates the causal chain.

time an inference is drawn increases. This leads to greater memory/storage requirements which could be an important factor in applications where the network may consist of a large number of highly interconnected nodes (large and possibly sparse CPTs for all the non-root nodes). It may also result in slower execution times for the probabilistic inferencing algorithms, as more data needs to be parsed through ([Liu, 2008] presents the use of a self-organizing map to address this issue very effectively).

(a)

State	Probability
A=0.5 => p(0≤ A ≤1)	0.3333
A=1.5 => p(1≤ A ≤2)	0.3333
A=2.5 => p(2≤ A ≤3)	0.3333

(b)

State	Probability
A=0.25 => p(0≤ A ≤0.5)	0.16666
A=0.75 => p(0.5≤ A ≤1)	0.16666
A=1.25 => p(1≤ A ≤1.5)	0.16666
A=1.75 => p(1.5≤ A ≤2)	0.16666
A=2.25 => p(2≤ A ≤2.5)	0.16666
A=2.75 => p(2.5≤ A ≤3)	0.16666

Figure 4-4: Discretization

Consider a discrete random variable A, such that $0 \leq A \leq 3$. Figure 4-4 shows two possible ways in which the CPT can be discretized. In each case, a uniform distribution has been assumed for A. If ε is the acceptable deviation from the steady state value $A=a$, then the bin limits are defined as $[a- \varepsilon, a + \varepsilon]$. The size of the bin is 2ε. As described in [Mehranbod, et. al. 2003], the smallest value of ε is dictated by the accuracy (β) of the sensor corresponding to A (i.e. $\varepsilon \geq \beta$ and in most cases, $\varepsilon = m\beta$, where m=1, 2, 3,…). It can be seen that the increments with which the value of A can be inferred using the network is much better in table (b) which has a greater granularity in discretization, as compared to table (a)[11]. Suppose the sensor for A (with an accuracy of ±0.05) indicates a value of A=2.125. Then according to table (a), this value falls in the bin $2\leq A \leq 3$. Hence, from the Bayesian network perspective the state A=2.5 will be considered as the value for node A (for instance, when the value of A has to be set as evidence during inference). However, according to table (b), the value indicated by the sensor falls in the bin $2\leq A \leq 2.25$, which results in a value of A=2.25. Thus discretization can have a significant impact on how closely the actual value of the variable (measured by a sensor) is represented in the

[11] It must also be noted that with the smaller discretization, the accuracy of the sensor used also needs to be higher. The effect of sensor accuracy is discussed in Chapter 5.

network for inference purposes and needs to be carefully determined by the domain expert who designs the network. Depending on the application, a large discretization may result in inferred values that are ambiguous for decision-making, especially if such values are intended for precision system control.

4.3.8. Sensor Reliability

Sensors can be affected by a number of factors in their operational environment. Factors like heat/temperature cycling, mechanical shock/vibrations, humidity, power-on/ power-off cycling, etc can sometimes have detrimental effects on the on-board signal processing electronics (for instance, oxidation and failure in solder joints, fretting leading to unreliable contacts, etc. [Fraden, 2004][Denton, 2010]). For sensors not based on a non-contact operating principle, the sensing element may itself undergo wear and tear due to physical contact. In most cases, the data from sensors is sent to a remote data acquisition device or a computer, where it is transformed into useful information (for instance, performance maps) that may be used for decision making. In this process, data from sensors may become unavailable due to a fault in intermediate connectors or wiring that conveys the sensor output signal to the processor (the analogous situation in case of wireless sensors would be a fault in the transmission link). Most sensors also need a power supply; a fault in the power leads may cause the sensor to become inoperative. All the factors described above may be taken together as representative of how reliable a sensor is.

[Fraden, 2004] defines the reliability of a sensor as its ability to perform its required function under specified conditions for a stated period[12]. Reliability is often expressed as the probability that the sensor will function without failure over a certain time or a specified number of cycles of use. A common metric for specifying reliability indirectly is in terms of mean time between failure (MTBF) which is the average

[12] Reliability in the present context is not purely a judgment on the quality of the sensor itself since it also encompasses other factors that may cause the sensor data to become unavailable nor does it refer to the extent to which the sensor output can be considered representative of the actual measurand. It only indicates whether or not data is consistently available from a sensor.

expected time between failures of like units under like conditions (as specified in the MIL-HDBK-217 standard) [Denton, 2010]. It is typically calculated based on installed equipment (MTBF = total time exposure for all installed units/ number of failures). Such information is rarely provided in the sensor specifications from manufacturers due to factors like the lack of a standard measure for reliability, the need for accelerated life testing under extreme environmental conditions, etc. [Fraden, 2004].

However, if such data is available for any system, for example, based on the operational history of the system and the various sensors integrated into it, the knowledge may be used to refine the structure of the Bayesian network for future versions of the system. The nodes corresponding to sensors which are traditionally found to be extremely reliable may be connected to as many other nodes as possible, representing other sensors which may be less reliable, in order to provide a greater assurance of back-up information being available in case of a loss of information from the unreliable sensors. In Figure 4-5 (a) and (b) suppose, the sensor corresponding to the node S_3 is considered to be the most reliable amongst all the available sensors. In case one of the sensors S_1, S_2…etc. becomes unavailable, then the network structures shown can help infer the value of those sensors using the value of S_3 within acceptable limits (depending on the quality of data used to generate the CPTs).

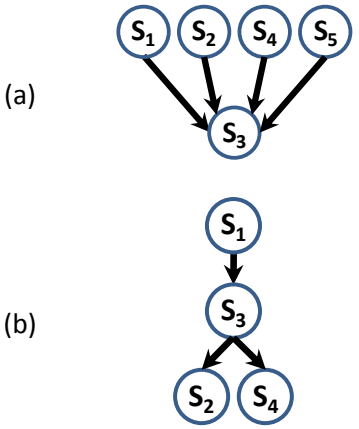


Figure 4-5: Sensor Reliability

4.3.9. Memory Requirements

By exploiting the conditional dependencies/independencies between the different random variables of interest (embedded explicitly in the network structure in the form of the links between the nodes corresponding to the variables), a Bayesian network allows compact storage of their joint probability distribution locally in the form of CPTs for all the non-root nodes in the network. As seen earlier, if data correlating all the variables is

available[13] then there are multiple ways in which the nodes may be connected based on the domain experts' opinion[14]. However, in doing so, caution must be exercised since different types of connections can result in different CPT configurations for various nodes. The resultant form of the CPTs can have a significant impact on the usefulness of the overall network in addressing the system's operational goals.

Consider two possible network structures that relate variables of interest in a system *A*, *B*, *C*, *D*. Assume that each variable has two states *True* or *False*. In Figure 4-6 (a) the total number of parameters in the CPTs[15] of *A*, *B* and *D* is 2 each whereas the number of parameters in the CPT of *C* is 4. In Figure 4-6 (b), the number of parameters in the CPTs of *A*, *B* and *D* is again 2 each but the number of parameters in the CPT of *C* is now 16. If one unit of memory is required to store each parameter, the total memory required in the first case is 10 units but increases to 22 units in latter case. With a more complex network, there may be several nodes with a large number of parents, a high degree of interlinking among the nodes, and a large number of individual states for each node. The size of the CPT for a node grows exponentially in terms of the number of parents. For a node with n states and i =1, 2, 3….k parents, if S_i is the number of states for the i^{th} parent, the size of the CPT for that node is n rows and $m= \prod_{i=1}^{k} S_i$ columns and the total number of parameters in the CPT is n x m. Thus, the size of the individual CPTs and the total memory requirements can spiral out of control quickly.

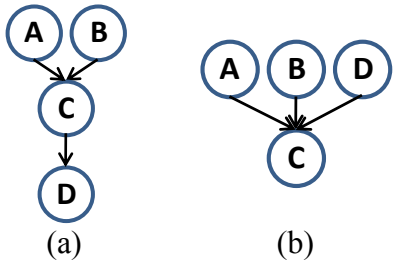


Figure 4-6: Memory Requirements

Even though the cost of memory/storage may not be expensive compared to the cost of other components in the system (and continues to decrease at a rapid rate), the on-board memory available for storing the CPTs may be limited due to factors like storage

[13] For variables $X_1, X_2, \ldots X_n$, for every value of X_i, a corresponding value of X_j ($i \neq j$) is also available
[14] This may not be applicable when causality is also taken into account in creating the network structure
[15] The number of parameters required to specify the conditional probability distribution of a node is said to be the size of the CPT for that node.

requirements for other programs/functions that are needed for effective system control and operation, for logging data for CBM, etc. Large CPTs can also prove to be a computational hindrance for applications where real-time operation is critical (due to the longer times needed to parse and extract values from the CPTs in inferencing algorithms, especially if the CPT is sparsely populated, or individual state probabilities are low and widely spread, etc.). The memory requirements must therefore be taken into account while designing the network[16]. Various techniques may be used to modify both the structure of the network (and the resultant size of CPTs as well memory required to store and manipulate them). These include the judicious selection of the number of levels of discrete states that are needed for every node in the network (especially for nodes which are connected to a child node with many other parent nodes), use of canonical models such as noisy-OR, noisy-MAX, etc. which reduces the number of parameters required to completely specify the CPTs [Diez and Druzdel, 2007], the introduction of intermediate nodes to 'divorce' parent nodes and partition their configurations [Jensen and Nielsen, 2007] which has the result of reducing the number of parent nodes associated with a given node, the use of decision trees or graphs, propositional rules (if-then), deterministic CPTs (with only 0 or 1 as probability values) [Darwiche, 2009], etc.

4.3.10. Intended Use

With the development of a variety of inferencing algorithms and advances in computational power, the use of Bayesian networks as a tool for both modeling and decision making has been increasing in many domains for objectives like diagnosis, fault detection and classification, etc. The extent to which a system is accurately represented by the model and the quality of results obtained using the model are direct functions of the network structure. For instance, [Speigelhalter, et. al., 1993] demonstrate that

[16] Although the size of the individual CPTs and the total size of the state space, as defined in terms of the total number of parameters (a similar approach is also found in [Jitnah and Nicholson, 1998]), gives a good idea of the network complexity and memory requirements, more sophisticated measures such as those discussed in [Gelly and Teatud, 2005] may be used by the domain expert while designing the network to decide on suitable network structures (or compare between alternative structures when the total number of parameters involved are the same) that can enable more efficient usage of available memory.

inferencing algorithms are as sensitive to the network structure as the probability values encoded in the different node CPTs. [Nadkarni and Shenoy, 2004] state that the most effective networks are those that combine sound expert knowledge to define the network structure (qualitative) and use extensive data to identify/refine the probability values of the variables represented by the nodes in the network (quantitative). However, despite the value of such a knowledge-based approach [Nadkarni and Shenoy, 2004], there is no prescribed method to construct the network structure when done by domain experts[17].

The process of creating the network structure based on expert opinion is iterative. A basic structure is first created and then refined based on feedback from other experts (often the direction of links that result from this process imply causality). Then, using the preliminary structure, the network may be implemented under real-world conditions (with components like a graphical user interface, visualization tools, etc. added) to carry out a particular task. This is done to verify its ease of use and intuitiveness in conveying the system characteristics to the end user. Based on user feedback, the network may once again be modified, if necessary, for better usability. If it is found that the results obtained using the network are not satisfactory (or worse, contradictory to those expected based on expert opinion or user experience), its structure may need further refinement. At each iteration, links or nodes may be added to the network or they may be pruned, the direction of some links may be reversed, etc. These small changes may or may not always be beneficial. In some cases, may possibly diminish the efficacy of the network in achieving its intended purpose (since each change may affect factors like the size of node CPTs, type of data / experimentation needed to estimate the CPT parameters, etc).[18]

---

[17] [Nadkarni and Shenoy, 2004] present one possible method to formalizing the process of construction of Bayesian networks by adopting a causal mapping approach

[18] When there are few domain variables involved it may be possible to systematically enumerate all the possible network configurations for example as presented in [Dean, 2006]. As seen in Chapter 2, the structure of a Bayesian network represents the Markov condition. Two networks may said to be Markov equivalent if they consist of the same set of variables (i.e. they represent the same joint probability distribution) and they represent the same conditional independence relationships between the variables [Korb and Nicholson, 2004]. The problem of then choosing the best possible structure to describe the dependencies between the variables represented by available data is referred to as 'model selection' [Neapolitan, 2003]. To do so, various scoring functions such as the Bayesian Information Criterion (BIC), Akaike's Information Criterion (AIC), Minimum Description Length (MDL), Bayes' factor etc. are used

Consider a case where a domain expert creates a network for a system with a set X of critical variables and a set Y of variables of secondary importance. In such a case, it would be imperative to represent all the variables in X as nodes in the network but the expert has to make subjective choices regarding how many/which specific variables from Y also need to be included in the network, if these variables are measurable, their relevance to the variables in X as well as to the goals of creating the network, etc. If such a network is intended to be used for real-time operation[19], then the insertion of numerous additional nodes into the network or a high degree of interlinking between the nodes in X and Y may render it too intractable to satisfy the real-time operation criterion (due to large CPTs resulting slower execution times for inferencing algorithms).

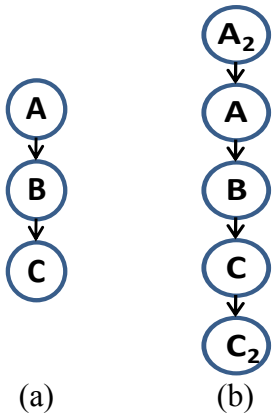


Figure 4-7: Use of Redundant Nodes

In other cases, it may be necessary to introduce additional nodes into the network to increase its effectiveness in achieving the application objectives. Consider the network in Figure 4-7 (a) designed for decision making in a condition monitoring application. Assume that each node in the network represents a sensor corresponding to a domain variable of interest and each link represents a physical process that transforms the variable represented by the parent node to the one represented by the child node. With any unexpected deviations in sensor readings, the challenge facing the decision maker who operates the system is to decide if the variations indicate a potential fault in one or more sensors or whether they are indicative of a fault in the monitored system. If the variations are inadvertently attributed to faulty sensors when in reality, they may be the result of degradation in one of the system's sub-components, it can result in a false alarm from the condition monitoring algorithm that utilizes this network (either calling for unscheduled system maintenance or in the extreme case, leading to complete catastrophic

---

are often used to especially when the structure/parameters of the network are learnt from extensive datasets [Korb and Nicholson, 2004]. [Kjaerulff and Madsen, A.L., 2008] [Speigelhalter, et. al., 1993].

[19] The definition of real-time being dictated by the end application

system failure). Section 5.4 presents the development of a novel Bayesian network based algorithm to detect and isolate the cause of such deviations (sensor vs. system/process). However, as will be seen in Section 5.5.4, the algorithm requires the addition of nodes (representing redundant sensors) to distinguish between sensor and system faults at extremities of the network (root and leaf nodes and the links attached to them). Even though the size of the network increases marginally, the addition of the redundant nodes is critical to achieve the desired functionality in the fault detection and isolation algorithm. Thus, the intended use of the network must always be taken into consideration while designing and before finalizing the structure of the network.

4.4. OPERATIONAL CRITERIA

Once the system design has been completed (with the requisite sensors integrated into the system) and a representative Bayesian network has been designed for it, the next step is to determine suitable criteria that may be used for managing information from all the sensors while the system is in operation. The objective is to make the best use of the information available from the finite set of sensors and the network in conjunction with the available computational resources at any given time. These operational criteria may be used to make decisions regarding how the available sensors may be prioritized to adapt to varying task demands, determine the best options for sensors that may serve as alternatives used to infer the value of failed sensors, what sort of information can be gleaned from the network, account for constraints that may arise during operation like limited bandwidth/power, decide on algorithms that are best suited to meet the application constraints, etc. The following section describes some such criteria (Note that these are general criteria applicable to many applications. They are intended to motivate the definition of more application-specific criteria along similar lines in the future.)

4.4.1. Sensor Characteristics

During the testing and certification process for any system, high quality data (i.e. with good discretization and low uncertainty) is used to generate equally exceptional performance maps over the entire operation range for all the variables of interest. The

CPTs for the nodes in the network are also derived from the same dataset. These as-built maps and CPTs represent the baseline standard against which any variation in those variables will be benchmarked when the system is deployed. Thus, they represent the best possible resource of knowledge as to how a particular variable is likely to behave under specific operating conditions.

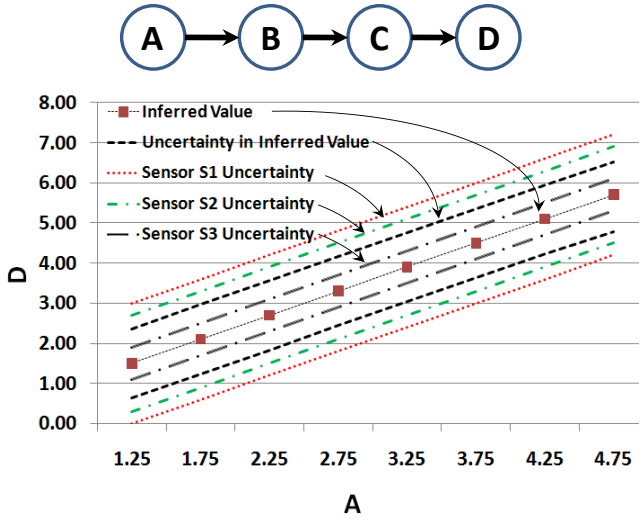


Figure 4-8: Performance Map Accuracy vs. Sensor Accuracy

The sensors used in the system certification process (the 'gold-standard' sensors) tend to be of better quality (better accuracy, resolution, good linearity, low noise, etc.) than some of those that are actually integrated into the system. This situation arises when based on certain design criteria (like cost, geometry constraints, etc. described in Section 4.3), a compromise is made with respect to the type or the quality of such sensors that are finally selected (or designed), to enable their seamless integration into the system. In some cases, the sensors may not perform equally well over the entire operating regime that the system is designed for. For instance, a high resolution incremental encoder may be used to calculate motor speed fairly accurately (by differentiating its output) at low to moderate speeds, but missed counts at may lead to significant errors in the calculated speed when the motor operates at high speeds. In such cases, it may be beneficial to use the Bayesian network to infer the value of the nodes corresponding to those variables, by using information from other sensors in the system as evidence[20]. If however, the sensor

[20] The sensors corresponding to these nodes of interest can then be used to corroborate the inferred values. In a way, they provide hardware redundancy for the inferred value. These sensors then are used as the primary source of information only in scenarios when the sensors used to set the evidence for inference themselves become unavailable and it is not possible to carry out the process of inferencing effectively.

corresponding to the node of interest is as good as the one used during certification, then the value of the variable corresponding to it would be determined using the sensor reading directly. The choice of which information source to use is made by the system operator.

Consider the network in Figure 4-8[21]. Suppose there are three types of sensors S_1, S_2 and S_3 (based on different operating principles) available for the variable represented by the node D. Assume that all three sensors satisfy the requirements in order to be integrated into the system, but each has a different level of accuracy. For instance, let the accuracy of the three sensors be ±1.5, ±1.2 and ±0.4 (with respect to the absolute values) respectively over the entire measurement range.

These values represent the uncertainty in the measured value of D when the readings are obtained directly from the sensors. Now, the value of D can also be obtained by using the reading from sensor corresponding to A, to set the corresponding state of A as evidence to the network and using probabilistic inference. From Figure 4-8, it is observed that the average uncertainty[22] over the entire range when the value of D is inferred in this manner is ±0.86 which is better than the accuracy obtained using the sensors S_1 or S_2. Thus, in this case, if either of these sensors is actually integrated into the system for measuring D, it may be a better option to infer its value using the network[23]. However, if S_3 is the sensor integrated into the system to measure D, then the value of D may be directly obtained from the sensor itself, given that it provides a better accuracy as compared to the inferred value.

---

[21] The range of the physical variables represented by the nodes A, B, C, and D are [1, 5], [2, 10] [1, 5] and [1.2, 8] respectively. For this hypothetical example, it is assumed that the variables can be related by simple analytical relationships of the form $B=2A+ \varepsilon_{A->B}$, $C= 0.5B+ \varepsilon_{B->C}$, $D= 1.2C+s \varepsilon_{C->D}$; $\varepsilon_{A->B}$, $\varepsilon_{B->C}$ and $\varepsilon_{C->D}$ represent the uncertainty in the processes $A->B$, $B->C$ and $C->D$ respectively. The probability distributions in the CPTs are generated using standard normal distributions with a standard deviation of 0.15.

[22] The values of all the nodes in the network are in the form of discrete distributions. For the present work, the term 'uncertainty' as used here refers to the spread /range of the states for a given node where the cumulative probability value of the marginal posterior distribution ranges from 0.05 to 0.95

[23] This would be feasible provided the system does not suffer any degradation in the long run, which in turn would render the maps/ CPTs involved in inferring the desired value inaccurate. The same caveat would also need to hold true for the sensors that are used to set evidence during the inferencing process.

4.4.2. Node Distance

Correlating all the variables of interest in the system using a Bayesian network allows the use of any variable to infer the value of any other variable in the network (by setting the former as evidence and using probabilistic propagation to infer the latter). However, the inferred value (and the uncertainty in it) can be heavily influenced by the number of intermediate links between the evidence node and the query node. Consider the network in Figure 4-9 (a). Suppose the sensor corresponding to node S_3 has failed but all the other sensors are operating correctly. Given the network structure, it is possible to use the data from any of the remaining sensors S_1 to S_5 to set a state of their corresponding nodes as evidence and infer the value of S_3. Intuitively, it can be expected that the uncertainty in the inferred value of S_3 will be the least when the value of S_2 is used as evidence as there is only one intermediate link between S_2 and S_3. In this case, the uncertainty in the inferred value is dictated by the uncertainty in the process S_2->S_3. This relation between S_2 and S_3 is encoded in the CPT of S_3 i.e. $P(S_3|S_2)$. Now, if the data from the sensor corresponding to the node S_1 is used to infer the value of S_3, then the final value is influenced by the uncertainties in two intermediate processes i.e. S_1->S_2 and S_2-> S_3. In this case, the value of the node S_3 will be calculated using the chain rule of probability as $P(S_3|S_1)= P(S_3|S_2).P(S_2|S_1).P(S_1)$. Since $0\leq P(.|.)\leq 1$, the value of $P(S_3|S_1) \leq P(S_3|S_2)$. In general, in the latter case, the probability distribution is spread over more states of the node S_3 with a lower probability value for each individual state. Thus, the farther the away the evidence node S_E is from the query node S_Q, the greater is the potential uncertainty in the inferred value of S_Q since each local inference introduces additional uncertainty/deviation in the final value[24]. This effect

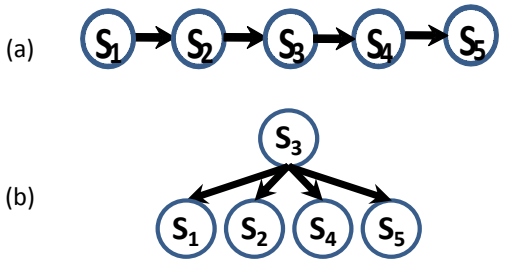


Figure 4-9: Node Distance

[24] The actual uncertainty will depend on the level of discretization, CPTs, etc. of the intermediate nodes between S_E and S_Q

may be quantified by using the concept of Node Distance (*ND*) for a single evidence node and query node.

Definition 4.1: Node Distance (*ND*) may be defined as the shortest possible path[25] between an evidence node S_E and a query node S_Q along a directed path between the two.

Using the notation $ND_{SE, SQ}$ the value of node distance can be calculated in terms of the number of intermediate links connecting the sequence of adjacent node pairs between S_E and S_Q. For instance, in Figure 4-9 (a) considering the nodes S_3 and S_5, the node distance is $ND_{S3, S5}$=2. Similarly, $ND_{S4, S5}$=1. As the value of *ND* increases, the greater is the potential uncertainty in the inferred value. This may be a guideline used by the system operator when determining which of the operational sensors may be used to infer an unavailable value.

However, caution must be exercised since the concept of *ND* may not work well for certain types of network structures. Consider the network in Figure 4-9 (b). Suppose the sensor corresponding to S_3 is determined to be faulty. Any of the remaining sensors may be used to determine the value of S_3. In this case, it must be noted that even though there is only one link connecting any of the nodes S_i, where $i=1, 2, 4, 5$ to S_3 (i.e. $ND_{Si, S3}$=1) the uncertainty in the final value of S_3 will be different depending on which of the nodes is used as evidence. In this case the uncertainty in the inferred value would be dictated by the uncertainty in the relations $S_3 \rightarrow S_i$ encoded in the respective conditional probability distributions i.e. $P(S_i|S_3)$. For such network structures, the concept of link strength (Section 4.4.6) is more suitable.

4.4.3. Sensor Health Status

The primary goal of integrating sensors into any system is to provide real-time feedback on the measurands of interest for control purposes and enable the system to successfully accomplish its task (for instance, a quality joint position sensor is crucial for a robot to achieve the desired positioning accuracy in high precision manufacturing

[25] Similar to the concept of geodesic distance in graph theory

tasks). An equally important task for both the essential and optional sensors in intelligent systems is to enable monitoring of variations in parameters over an extended time by providing reliable and accurate data to periodically update the relevant performance maps. The goal is to track the overall health of the system using condition based maintenance algorithms [Hvass and Tesar, 2004] [Shakkari and Tesar, 2008] to ensure continued availability of the system as well as to assist the human decision maker in determining the ability of the system to accomplish the required tasks. The implicit assumption for the above objectives is that all sensors are operating as per their design/operational specifications and the data obtained from them is always dependable.

A sensor can be considered 'healthy' if it produces an output signal proportional to the input stimulus, within an acceptable amount of deviation as dictated by the sensor physics, resolution, accuracy, application requirements, etc. However, as mentioned earlier, the output from the sensors can be affected during regular operation by a number of factors. The effects of these are manifested as undesirable deviations in the sensor output like drift, bias, excessive signal noise, etc. Such phenomena may be considered as faults in a sensor that occur intermittently or they may occur consistently over an extended period indicating the development of gradual sensor faults. In the extreme case, there may be a complete loss of information from a sensor due to an abrupt failure of the sensing element or its peripherals like power/signal transmission lines, connectors, faults in the onboard signal processing circuits, etc. When the required sensor readings become unavailable or when erroneous sensor readings are used for control purposes, it may lead to undesirable system behavior.

Furthermore, using data from faulty sensors to update performance maps, without checking for their validity will result in corruption of the stored maps. This in turn, may lead to false alarms and missed detection of system faults from the system-level CBM algorithms. In each situation, the health of all the sensors must therefore be taken into account by the system operator in deciding whether or not to utilize the data from a particular sensor. To this end, Chapter 5 presents the development of a novel Sensor and Process Fault Detection and Isolation (SPFDI) algorithm that can help quantify the

trustworthiness of the information from a sensor. This is achieved by correlating, via a Bayesian network, the information from all the sensors monitoring the system and calculating an intuitive metric representing the health of each sensor, that the decision makers can then use in their assessment.

4.4.4. System Health Status

An unanticipated deviation in a sensor's readings from its expected values under specific operating conditions may not necessarily be due to a fault in the sensor itself or in any of its peripherals for that matter, but may be indicative of a more serious, potential fault in the monitored system itself. For instance, consider an electric motor used in a harsh environment and monitored using accelerometers and a torque sensor as optional sensors, in addition to position, current and voltage sensors as the essential sensors for control and commutation. In such applications, there may be a high potential for dirt and other debris entering into the motor shaft bearings and causing damage to the rolling elements within a relatively short duration of time. A damaged bearing within the motor may lead to significant ripple in the output torque as well as excessive vibrations and acoustic noise. The effect of the flawed bearing would thus be observed as large spikes in the readings from the accelerometers and the torque sensor. Although such a fault would be eventually detected by the higher level CBM algorithms monitoring the motor, such a diagnosis would be obtained only after a more extended period of monitoring, by which time the fault may have seriously affected the system capability and life. Without any knowledge of the system status, the unusual sensor readings may wrongly be construed as potential faults in the associated sensors (for instance, if simple limit checking is used to validate the sensor readings). Hence it is necessary to make at least a preliminary assessment of the system's health status, based on the sample of data available at a given instant from all the sensors monitoring the system.

A solution to this issue is again provided by using the Bayesian network framework to model the system and correlate the measurands. The SPFDI algorithm presented in Chapter 5 can not only provide a quantitative measure of a sensor's health

(to enable a correct judgment by the operator in deciding whether or not to utilize the data available from the sensor), but can also provide a similar quantitative measure for the various processes that occur in the system. These processes, symbolized by links in the network, represent the roles played by the different components of the system in the transformation of one variable to another. Hence, any undue change in a particular process can be deemed to be indicative of a potential fault with the components of the system associated with that process. The challenge here is to distinguish if the deviations in the sensor readings are due to a change in the nature of one or more processes or if it is the result of flawed sensors. If it can be determined with some degree of belief that the cause of undue variation in the sensor readings is a process in the system and not the sensor, this can help alert the system operator to monitor the outputs from the system level CBM algorithm more closely to determine the system availability.

4.4.5. Resource Availability

In most applications, following some preliminary processing at the sensor-level, the signals from all the sensors monitoring the system are sent to a central location for further processing or for use in deriving higher level information. This configuration is commonly observed in PC-based data acquisition and control of systems like EMAs, mobile robots, etc. With a limited number of sensors, a point-to-point connection technique is sufficient to connect the sensors directly to the PC without significant design or hardware overhead. However, such an arrangement requires complex cabling arrangements. Hence a bus topology is often utilized wherein all the sensors use a common set of resources for data transmission [Frank, 2000]. In a digital fieldbus system, multiple sensors are connected via shared digital communication lines (thereby reducing the number of cables) to transmit/receive data more efficiently on an as-needed basis [Potter, 1998][Pal and Rakshit, 2004] [Pitzek and Elmenreich, 2003]. When such an arrangement is utilized, the cumulative data bandwidth and latency[26] required for all the

[26] The term bandwidth here refers to the amount of sensor data that is transmitted in a specified time period whereas latency refers to how quickly the data transfer occurs.

sensors being considered plays a significant role in the selection of the appropriate bus[27]. This is largely dictated by factors like the type of the sensor output, the quantity of output data generated in a specific time period, the sampling rate used for the different sensors, mode of acquisition from multiple sensors (simultaneous/multiplexed), etc.

Consider for example, a motor equipped with an incremental encoder producing 10,000 counts per revolution (cpr) and rotating at a moderate speed, say, 600 rpm. This yields an output signal frequency of 0.1 MHz[28]. As the motor speed increases, the volume of output data from the encoder also increases. In addition, the motor may be instrumented with other sensors like current, voltage, temperature, etc. which may generate additional volumes of data. To acquire all this information accurately, it needs to be sampled at a high rate[29]. Hence, in addition to the transmission bandwidth, the data acquisition hardware also needs to be capable of handling the frequency requirements for sampling.

With fewer sensors, the total bandwidth requirements are moderate and it may be possible to sample all the sensors simultaneously with the available data bus and acquisition hardware resources. However, if the system has a large number of sensors which also need to be sampled at high rates, the number of high-speed data acquisition channels required increases (to accommodate the increased bandwidth/sampling requirements) which typically leads to a higher overall costs. Often as a compromise between cost and performance requirements, a limited number of data acquisition channels are used (capable of handling large amounts of data at high frequencies) and the available resources are distributed across all the sensor channels, by using a lower sampling rate, polling the sensors periodically instead of continuous acquisition, etc.

---

[27] A variety of sensor level communication buses, each with different bandwidth and data transmission capabilities, are used in industry. These range from serial interfaces such as RS-232, RS-422 and RS-485 to more advanced buses such as PCI, PXI, USB, Actuator Sensor-interface (AS-i), Local Operating Network (LON) [Pratt, 2004], Controller Area Network (CAN) [Zhang et al., 2004], Bidirectional Synchronous-Serial sensor interface (BiSS) [Quasdorf, 2003], Serial Peripheral Interface (SPI) [Zhou and Mason, 2002], Highway Addressable Remote Transducer (HART) protocol [Helson, 2004], etc.

[28] Based on $f = [(\text{cpr}) \times (\text{rpm})]/60$

[29] To accurately measure this data, it needs to be sampled at a rate greater than at least twice the highest frequency of interest, based on the Nyquist sampling criterion. This is done to prevent phenomena like aliasing where false frequency components may appear in the sampled signals

The use of a Bayesian network to model the system allows the flexibility of inferring value of any node/variable in the network (query) using the value of any other node/variable (evidence) in an inferencing process (Section 2.6). This capability can be exploited for managing the available resources (bandwidth/sampling rate capability) in certain operating regimes of the system, where it may not be possible to accurately acquire data from sensors with demanding requirements (i.e. those that require a high bandwidth/sampling rate). For instance, in the example cited earlier, if the motor rotates at 6000 rpm, the output frequency from the encoder rises to 1 MHz. If the associated data bus and acquisition hardware are capable of accommodating only 0.5 MHz, it might be more prudent to allocate the available resources to sensors with modest resource requirements, say, the voltage sensors which need to be sampled at only 1kHz to acquire their output data with the best possible resolution/sampling rates. This data may then be used to infer the values of other variables that have higher bandwidth/sampling rate needs such as motor speed and position (within reasonable accuracy) using a Bayesian network that includes the motor terminal voltage, speed and position as nodes[30].

4.4.6. Strength of Relationship between Nodes

The structure of the Bayesian network explicitly represents the conditional dependencies/independencies between the different variables of interest in the system (nodes). The strength of these conditional relationships is encoded in the conditional probability parameters of the CPTs for all the non-root nodes in the network. However, in any system, a particular set of physical variables, say X, may have a greater influence on a set of variables Z than another set of variables Y. For instance, consider the effect of the components of the magnetic flux density B_r (radial) and B_t (tangential) on the torque generated by the switched reluctance motor (Section 3.3.5). Both the components have a combined effect on the production of motor torque. However, B_t has a greater contribution as compared to B_r towards producing useful motor torque whereas the radial

[30] The caveat in using such an approach is that it assumes that the relations between the different variables as encoded in the parameters of the node CPTs (which are created from high quality performance maps data) remain the same even when the motor operates at high speeds.

component B_r has a parasitic effect in that it contributes more towards the generation of acoustic noise rather than usable motor torque (Section 3.3.4). In such cases, in the scenario that information one or more sensors corresponding to the variables in Z becomes unavailable, it would be desirable to use the information available from the sensors corresponding to the variables in X rather than in the set Y, in order to infer the values of the variables of interest in the set Z.

An approach using which the extent of such influence may be quantified is by using the concept of link and connection strengths. These measures were first introduced by [Boerlag, 1993] for Bayesian networks with binary nodes (two states). The connection strength measures the strength between any two nodes in the network (without accounting for the path between the two) whereas link strength (also referred to as arc weight by [Nicholson and Jitnah, 1999]) specifically calculates the strength along a particular link between two adjacent nodes[31].

Both these ideas were introduced initially as a way to improve the visualization of the network structure when learnt from data (for instance, using thicker links to represent stronger relationships) but were also later used to improve the efficiency of inferencing algorithms (for instance, by eliminating links arcs with insignificant weights) [Nicholson and Jitnah, 1999] [Ebert-Uphoff, 2007]. The connection and link strength are based on the information theory concepts of entropy and mutual information. The entropy and conditional entropy of a discrete random variable are given by [Ebert-Uphoff, 2007] as follows:

$$U(A) = -\sum_{a_i} P(a_i)\, log_2 P(a_i)$$
$$U(B|A) = -\sum_{a_i} P(a_i)\, U(B|a_i) \qquad \text{Eq. (4-1)}$$

[Ebert-Uphoff, 2007] defines the connection strength between any two nodes/variables A and B in the network, as how strongly the knowledge of the state of A

[31] [Jitnah and Nicholson, 1999] extended the basic idea of arc weight to include measures like the weight of a connected set or a cluster of nodes, the weight of a chain of nodes, the weight of a loop, etc.

affects the state of B and vice versa and quantifies it using the concept of mutual information as follows:

$$CS(A,B) = I(A,B) = U(A) - U(B|A) = \sum_{a,b} P(a,b) log_2 \left(\frac{P(a,b)}{P(a)P(b)} \right) \qquad \text{Eq.(4-2)}$$

The link strength is defined specifically for the relation A->B (i.e. A is the parent and B is its child). If C represents the set of other parents of B where C=$\{C_1, C_2 \ldots C_n\}$ and $\boldsymbol{c}$ represents the set of states of all the nodes C_i, then the link strength is defined as [Nicholson and Jitnah, 1999][32]

$$LS(A \rightarrow B) = \sum_{\boldsymbol{c}} P_{pr}(\boldsymbol{c}) \sum_{a} P_{pr}(a) \sum_{b} P(b|a,\boldsymbol{c}) log_2 \frac{P(b|a,\boldsymbol{c})}{P_{pr}(b|\boldsymbol{c})} \qquad \text{Eq.(4-3)}$$

where P_{pr} is an approximation of the prior probability of the node being in a particular state and is approximated by averaging the conditional probabilities of that node over all its parent state combinations.

For any application, the values of link strengths and connection strengths may be calculated between different sets of variables and used to determine the most appropriate sensors to use (i.e. if the corresponding nodes have high link/connection strengths indicating that the associated variables are strongly correlated) to infer the information corresponding to faulty or degrading sensors.

4.4.7. Type of Query

The Bayesian network compactly represents the joint probability distribution of all the variables represented by the nodes in the network. In other words, the network structure and the CPTs for the different nodes, represent a comprehensive database that can be queried in different ways to obtain different types of information regarding the system and its sensors. Depending on the application and the operating regime of the system, choosing the right type of query can provide information that is of greater value to the system operator for decision-making under the given circumstances. [Pearl, 1988] classifies these different types into two broad categories as belief updating and belief

[32] [Ebert-Uphoff, 2007] refers to this measure as the 'true average link strength'

revision. In belief updating, a measure of belief is associated with each individual proposition for every variable in the network and based on the available evidence, whereas in belief revision, the goal is to identify from each variable of interest, a composite set of propositions that can best explain the available evidence [Pearl, 1988]. The output of a belief revision process is thus a set of instantiations of the query variables, a list that may change as more and more evidence becomes available [Pearl, 1988]. Thus, belief updating provides a local summary of the network whereas belief revision provides a more global explanation of the network.

For any Bayesian network, it is possible to define four different types of queries as stated in [Darwiche, 2009]: probability of evidence, prior and posterior marginal distributions, Maximum Aposteriori Hypothesis (*MAP*), Most Probable Explanation (*MPE*). The probability of evidence query refers to the probability of an instantiation *e*, for a set of evidence variables *E* and given as $P(E=e)$ [Darwiche, 2009]. Such a query may be used to address the duty cycle characteristics of the variables of interest (typically, these are the control variables) in the system. For instance, in the Bayesian network shown in Figure 3-2, this query may help answer operator questions of the type 'what is the probability of the PWM duty cycle being 5 %'.

For a joint probability distribution of *n* variables, given by $P(X_1=x_1, X_2=x_2, \ldots X_n=x_n) = P(x_1, x_2, \ldots x_n)$, the marginal distribution is a projection of this distribution on a smaller set of variables (i.e. $P(x_1, x_2, \ldots x_m)$ where $m \leq n$) and is given by [Darwiche]:

$$P(x_1, x_2, \ldots x_m) = \sum_{x_{m+1} \ldots x_n} P(x_1, x_2, \ldots x_n) \qquad \text{Eq.(4-4)}$$

In belief updating, the most common type is the posterior marginal query $P(Q=q|E=e)$ where *Q* represents one or more query nodes whose values are to be determined and *E* is the set of evidence nodes whose values are observed and instantiated to the values e[33]. This value can be calculated by modifying Eq.(4-4) as follows [Darwiche, 2009]:

[33] If the marginal distribution is calculated in the absence of evidence, it is called a '*prior marginal*'.

$$P(x_1, x_2, \ldots x_m|e) = \sum_{x_{m+1} \ldots x_n} P(x_1, x_2, \ldots x_n|e) \qquad \text{Eq.(4-5)}$$

For instance, in the network shown in Figure 3-2, such a query may help answer operator questions of the form 'what is the probability of the motor producing 1.2 Nm torque for a PWM duty cycle of 5 % and PWM frequency of 10 kHz?'; i.e. the query enables the calculation of $P(T{=}1.2 \mid DC{=}5, FR{=}10)$. Such queries are important when the operator is concerned only about a specific physical variable in the network for decision making.

For a set of variables X, the *MAP* query determines the most probable instantiations for a specific set of query variables Q (where $Q \subseteq X$), given the evidence $E{=}e$ and is defined as [Darwiche, 2009]:

$$MAP(Q, e) = argmax_q P(q, e) \qquad \text{Eq.(4-6)}$$

where, $argmax_q\ f(q)$ indicates the assignment of values of q for which $f(q)$ is maximal (note that this is not the same as maximizing the posterior marginals of individual nodes). However, it must be noted that there may be more than one such possible set of instantiations which has the maximum probability. In Figure 3-2, such a query may help answer operator questions of the form 'what level of torque and acoustic noise level is the motor likely to produce for a PWM duty cycle of 5 % and PWM frequency of 10 kHz?'. *MAP* queries are important when the operator is concerned about multiple objectives simultaneously and needs to judge if the behavior of a specific group of variables in the network is consistent with other available knowledge for the purpose of decision making. For instance, in an application such as an EMA used in submarines, the system operator may be concerned with commanding the EMA to generate maximum torque while producing minimum acoustic noise at the same time. Based on other measurements like current, voltage, etc. the operator may be able to decide using the MAP query on a Bayesian network of the system if the projected torque and noise values are plausible.

The *MPE* query is a special case of the *MAP* query, where the goal is to determine the most probable instantiations for all the non-evidence variables (i.e. the query variables are all the unobserved, non-evidence variables). In other words, given a set of evidence values $E{=}e$, the objective is to find a consistent set of values for all the other

variables in X i.e. $\{X_1=x_1, X_2=x_2,\ldots X_n=x_n\}$ which represent most probable instantiations of those variables. The instantiation $x_1, x_2,\ldots x_n$ is then referred to as the most probable explanation given the evidence e. The *MPE* query can be defined as [Darwiche, 2009]:

$$MPE(X,e) = argmax_x P(x,e) \qquad \text{Eq.(4-7)}$$

Again, there may be more than one set of instantiations x that maximize the posterior probability. In general, *MPE* queries are simpler to determine algorithmically than *MAP* queries [Darwiche, 2009][34]. *MPE* queries are important when the system operator would like to determine the most probable behavior for all the variables of interest in the system, considered simultaneously, given the knowledge of specific variables (typically the control variables) in the network and examine if they are consistent with the available information for decision making. This may be important in a task like updating multiple performance maps for CBM of the different components in an EMA based on limited sources of measurable information.

4.4.8. Computational Complexity

As seen in Section 4.3, when a Bayesian network is developed for any application, the resultant topology depends completely on which criteria are accorded importance for that application. For very large or complex systems like aircraft, it may be simpler to design separate networks for each subsystem and analyze them separately, to avoid an overwhelmingly complex network structure for the whole system. When software like AMOS (Appendix C) is developed for Bayesian networks based decision-making [Ashok and Tesar, 2007], a diverse suite of inferencing algorithms is therefore typically incorporated into it for utmost flexibility in handling different network topologies and other application-specific operational constraints.

---

[34] Although the inference algorithms presented in Chapter 2 may be used to compute *MAP* instantiations (by computing posterior marginals and selecting the instantiation with the maximum probability), as described in [Darwiche, 2009], such an approach has exponential complexity as the number of *MAP* variables increases. For a detailed description of the different algorithms that may be used to calculate the *MAP* and *MPE* instantiations, the reader is referred to [Darwiche, 2009] (Chapter 10).

For a simple casual chain (Figure 4-10 (a)), the values for different nodes may be calculated by repeated application of Bayes' theorem. For polytrees (Figure 4-10(b)), exact inferencing may be performed by local computations and message passing between nodes (Pearl's algorithm). For multiply connected networks (Figure 4-10(c)), exact inferencing may still be applied by using clustering, conditioning, variable elimination, etc.

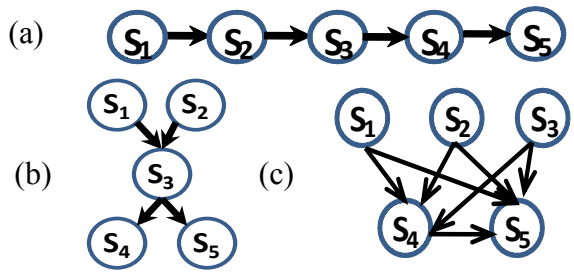


Figure 4-10: Computational Complexity for Different Network Structures

Although exact algorithms provide the greatest precision, applying such algorithms to even moderately complex or densely connected networks may require an exorbitant amount of computation/time and memory or in some cases may be unable to even complete [Cheng and Druzdel, 2000]. In such cases, approximate inferencing is used. Approximate algorithms (stochastic simulations, model simplification, etc.) provide a tradeoff for speed against precision and for very complex networks may be the only viable option. Stochastic simulation methods like Likelihood Weighting generate an imprecise answer quickly and refine it iteratively. The accuracy of results obtained from such algorithms generally improves as the number of samples generated increases[35].

In some applications, real-time operation is a binding constraint. In such cases, the utility of a computed result is estimated not only on the basis of its accuracy but also on the timeliness with which it is obtained. The utility is considered to degrade (even if the result is highly accurate) as time progresses beyond a predetermined deadline (application-specific) [Guo and Hsu, 2002]. The extent to which such real-time constraints can be satisfied is, in turn, dictated by factors like choosing the most suitable algorithm for a particular network topology (for instance, both exact and approximate

[35] For instance, if the *KL* divergence is used as a measure of convergence, then its value decreases as the number of samples increases, see Section 2.5.2.3

algorithms may be used for the structure in Figure 4-10(b)), the granularity of discretization for the different nodes (for instance, see [deSantana, et. al. 2007]), the efficiency of implementation (coding) of different algorithms in the operational software (optimized implementations with efficient coding tend to have faster execution times), etc. An important factor in most cases are the computational resources (time/space) available to execute these algorithms[36].

Though stochastic sampling algorithms are largely impervious to network size or topology, the number of samples that need to be generated to achieve the desired accuracy while satisfying real-time operational constraints may still be a challenge for large networks (the execution time tends to be directly proportional to the number of samples as well as the number of nodes and their interconnections)[Cheng and Druzdel, 2000]. For laboratory test set-ups, it may be possible to implement inferencing algorithms on computers with fast, dedicated processors and ample memory. Alternatively inferencing code may be implemented via parallel processing on multiple processor cores or distributed/cloud computing. In such cases, it may be feasible to provide real-time operation even for very large networks with highly discretized nodes. However, the same may not be true if the algorithms have to be executed on a system with restricted/ shared computational resources (for instance, a mobile robot where the available on-board computation resources have to be shared and allocated between many navigation, control, and communication functions). To account for such scenarios, the decision maker may use one of the many approaches described below to decide on an appropriate compromise between the available computational resources and the operational requirements.

Stochastic sampling algorithms are considered to be anytime algorithms. They can be interrupted at any point in time to yield an approximate result. If these are the algorithms of choice based on the network topology, then the decision-maker may vary the number of samples needed (based on suitable convergence criterion like the *KL* divergence) to satisfy the real-time execution constraint at the expense of some loss of

[36] The available computational resources and the resulting time/space tradeoff in the efficiency of algorithm execution can also be considered as an important criterion that may be used while designing the network topology (in addition to others described in Section 4.3)

accuracy. Alternatively, the decision-maker may also use domain characterization metrics as proposed by [Jitnah and Nicholson, 1998] as a basis to compare the execution characteristics of different inferencing algorithms on an existing network structure to aid in the selection of the most appropriate one. These include metrics for individual nodes (CPT skewness, maximum and average distance of a node from other nodes, node distance from query nodes) as well as for the entire network (size of state space, number of nodes and links, maximum and average state space size, connectedness = #arcs/#nodes, maximum and average CPT size, maximum and average number of parents etc.). The metrics may be used to generate a look up table beforehand characterizing the performance of the available suite of algorithms and then used by the decision-maker to select the most appropriate algorithm based on the task requirements at any given time[37]. A similar approach to prior offline compilation of algorithm execution profiles and their associated mapping to domain characteristics is also proposed by [Borghetti, 1996]

Other techniques may also be used to adaptively allocate the available computing time/memory when the system is in operation. Techniques such as recursive conditioning [Allen and Darwiche, 2003] provide an any-space approach (where the performance of an algorithm improves with increasing memory beyond a minimum requirement) to exact inferencing by utilizing conditioning to decompose a network into smaller sub-networks that are again solved independently and recursively. [Ramos and Cozman, 2005] provide another alternative approach termed as the adaptive conditioning framework to provide a tradeoff between time, space and quality of results obtained by decomposing the network into sub-networks and allocating different exact/approximate algorithms to process the sub-networks.

4.5. CHAPTER SUMMARY

With the use of multiple sensors to monitor a system like an intelligent EMA, there is a clear need for a criteria-based resource management framework that can help

[37] This may also be represented as performance map (for a specific algorithm like Likelihood weighting). For example, a map may be generated with algorithm execution time, memory required and state space size of network as the three axes.

decide upon and best utilize the available sensing resources at any given time, without compromising on the overall system performance.

This encompasses facets like deciding on the sensors needed to respond to changes in the monitored system and its environment, correlating multiple sources of information in the best possible manner, providing redundancy to ensure constant availability of information (under partial or complete sensor failures) and adapting the usage of available sensing and computational resources to changing task requirements while concurrently keeping the costs and complexity involved to a minimum. Based on the choice of a Bayesian network based methodology to address the above issues, this chapter explored some criteria that could be utilized to improve both the design as well as operational use of such networks.

To enable the domain expert involved in designing the Bayesian network for a system to develop a compact, more usable and efficient network topology, a preliminary list of ten design criteria were described in the different sub-sections of Section 4.3 and are summarized in Table 4-2. More such criteria may be defined for future applications based on the domain-specific requirements.

Once a representative Bayesian network for a system has been formulated (with the requisite sensors integrated into the system), to assist the system operator make the best use of the available set of sensors and network topology in concert with the available computational resources (prioritizing sensors to adapt to varying task demands, accommodate failed sensors, limited bandwidth availability, inferencing algorithms best suited to meet operational constraints, etc.), a preliminary list of eight operational criteria were discussed in the different sub-sections of Section 4.4 and are summarized in Table 4-3.

Two of these criteria (viz. sensor health status and system health status) will be explored in greater detail in Chapter 5. As in the case of design criteria, numerous other application-specific operational criteria may be defined in the future.

Table 4-2: Design Criteria (for a system and its Bayesian network representation)

No.	Criterion	Description /Importance	Section
1	Application Requirements	Any application environment consists of sensors that are essential for operation as well as additional optional sensors. The total number of nodes in the Bayesian network is dictated by the number of sensors used in any application.	4.3.1
2	System Design Constraints	The number of sensors used in any application is dictated by the form factor of available sensors, cabling, etc. that satisfy the size and volume constraints imposed by the application. These design constraints thus play a vital role in determining the size of the Bayesian network by influencing the sensors that can be actually used in the application.	4.3.2
3	Overall Costs	The overall costs associated with sensors and peripheral equipment like data acquisition hardware, power supplies, etc. may influence which sensors are integrated into a system and hence influence the number of nodes in the Bayesian network representation	4.3.3
4	Relative Importance of Sensors	The relative importance of a sensor or a subset of sensors in any application may be used to determine the direction and number of the links between the nodes. This allows the manipulation of network topology to provide the desired degree of redundancy for the most critical measurands.	4.3.4
5	Type of Data Available	The availability of data in specific formats (for instance, due to experimental constraints) influences the manner in which the conditional independencies between variables can be represented (i.e. which nodes in the network may be linked as well as the direction of the arcs) and how the CPTs for different nodes can be populated	4.3.5
6	Causality	Causal relations between physical variables can be used to determine the node ordering, the direction of links and the level of interconnectedness in a Bayesian network. The use of causality can result in more practical, intuitive, physically relevant and compact networks.	4.3.6
7	Discretization of Data	The number of distinct states into which each node in the network can be discretized is driven by application requirements. The granularity of discretization in turn influences the quality of results obtained using the network, speed of execution of inferencing algorithms, memory required to store the CPTs, etc.	4.3.7
8	Sensor Reliability	All the sensors are not equally reliable (different susceptibility to temperature, shock/vibrations, etc.). The operational history for sensors may be used to direct links from nodes representing the most reliable sensors to ensure constant availability of information in case of a loss of sensors corresponding to critical measurands	4.3.8
9	Memory Requirements	The size of CPTs grows exponentially in the number of node parents, degree of interlinking and the granularity of discretization of node states. The associated memory requirements must be considered to satisfy constraints on availability of resources (storage), real-time operation requirements, etc.	4.3.9
10	Intended Use	Modifications such as adding additional nodes or links (or pruning existing nodes/links) in a network structure may be needed to satisfy specific requirements for the application (for example, fault detection and isolation), provide greater intuitiveness to the end-user, etc.	4.3.10

Table 4-3: Operational Criteria (for a system and its Bayesian network representation)

No.	Criterion	Description / Importance	Section
1	Sensor Characteristics	Depending on the accuracy of the sensor used for a specific measurand, the operator may choose to use the value of sensor reading directly or infer the value of the measurand via the Bayesian using the readings from the other sensors as evidence.	4.4.1
2	Node Distance	The uncertainty in the inferred value of a variable is influenced by the number of intermediate nodes/links between the evidence node/s and the query node. In general, the greater the separation, the greater is the uncertainty in the inferred value.	4.4.2
3	Sensor Health Status	With any unforeseen variations (drift, bias, excessive noise) in the readings from a sensor or a subset of sensors, the system operator needs to be able to determine if the readings are indicative of one or more faulty sensors or if it is the result of normal changes in the system response	4.4.3
4	System Health Status	Unanticipated deviations in sensor readings may not necessarily be due to a faulty sensor but may be indicative of a more serious, potential fault in one or more of the monitored system's sub-components. A preliminary assessment of system health is therefore needed to enable the system operator to make this distinction.	4.4.4
5	Resource Availability	Different sensors may need to be used in different system operating regimes (to provide the value for a specific variable via direct measurement or by inferencing using the Bayesian network) in accordance with the bandwidth, sampling rates, etc. that are possible with the available system hardware	4.4.5
6	Strength of Relationship between Nodes	Some variables/nodes in the network may be more strongly correlated (or may have a direct causal influence) as compared to others. The strength of these relationships (quantified using the concepts of link strength and connection strength) may be used to determine which set of nodes are more appropriate for use as evidence for a particular set of query variables.	4.4.6
7	Type of Query	The topology of the Bayesian network along with the conditional probability distributions of the different nodes represents a comprehensive database for a system which can be queried in multiple ways (probability of evidence, prior and posterior marginals, *MAP*, *MPE*) to obtain different types of information regarding the system and the sensors monitoring it.	4.4.7
8	Computational Complexity	Based on the network topology, discretization of the nodes, available computational resources, real-time operation requirements, etc. the operator may decide on the appropriate exact or approximate inferencing algorithm to use, number of samples to be used for stochastic sampling (approximate) algorithms, etc.	4.4.8

Chapter 5. Sensor and Process Fault Detection and Isolation

5.1. INTRODUCTION

The benefits of employing performance maps and envelopes for human-directed decision-making in complex systems[1] with multiple inputs and multiple outputs (like industrial and mobile robots, aircraft, electric vehicles, manufacturing cells, etc.), have been described in-depth in [Tesar, 2009][Ashok and Tesar, 2007][Yoo and Tesar, 2006]. The system level controllers as well as the human operator, who makes the final decisions regarding the system operation, rely on these maps and envelopes to guide their choices. The maps are also used to continuously track the health of the system for condition based maintenance [Hvass and Tesar, 2004] and hence need to be periodically updated (ideally in real-time) in order to accurately reflect the extant operating conditions. To achieve the aforementioned capabilities, often a suite of multiple sensors is integrated into such systems in order to track the many physical parameters associated with regular operation. The use of any erroneous data in the updating process may result in corruption of these maps, which in turn may lead to incorrect operational decisions being made by the operator or result in false alarms being raised during condition monitoring. It may also result in unpredictable system behavior with potentially damaging consequences.

Accurate and dependable sensor data is thus of vital importance. With any unexpected deviations in the sensor readings from their theoretical or predicted values, the challenge is thus to determine if it simply indicates a change in the operational state of the system, if they are the result of an anomaly in any of the sensors or if they are indicative of a more serious issue in the monitored system itself. If the source of variation in the sensor readings can be determined with some level of certainty, then it is possible to quantify and incorporate this knowledge to provide additional guidance in decisions like whether or not a particular sensor can be considered operational, if a particular sensor is trustworthy enough for updating the maps or inferring other related parameters,

[1] In this report, the system refers to an intelligent EMA

whether the source of variation can be localized to a particular component of the system, etc. The Bayesian causal network framework developed thus far allows the different system parameters to be correlated in a physically meaningful way, while also explicitly accounting for the uncertainties in each of them. In addition, the framework also allows the use of prescribed methods to quantify, propagate and combine these uncertainties. All these characteristics can be harnessed to achieve the objectives mentioned above. Section 5.2 aims to provide the motivation for fault detection and isolation. Section 5.3 discusses the types of faults in the context of the present work and outlines the typical steps needed for any fault detection and isolation procedure, followed by a review of different existing approaches. In Section 5.4, a novel procedure based on Bayesian networks is presented to help distinguish between deviations in readings caused by faults in the sensors and those caused by faults in the monitored system. Sections 5.5 and 5.6 demonstrate the application of the algorithm using different examples, including some special cases which require specific modifications to the algorithm. Section 5.7 highlights some factors which may affect the working of the algorithm and its final results. Finally, Section 5.8 briefly addresses the significance of the algorithm results in updating the available information regarding the system operation (i.e. performance maps/decision surfaces).

5.2. SENSOR ISSUES

Sensor outputs are based on specific changes in the physical/ electrical properties of their primary sensing elements, proportional to a physical input (for instance, a change in the resistance of strain gages in response to load). Typically, most sensors produce an analog voltage or current output. Inbuilt circuits in the sensors or in data acquisition systems have some form of signal conditioning, A/D conversion, digital signal processing, etc. that digitizes this output for use by the system controller. Although the integrity of data finally obtained depends on each of the above steps, with most state of the art sensors, the final signal is a realistic representation of the original signal.

However, any change in the characteristics of the sensing element itself (due to wear, aging, etc.) may cause the sensor readings to deviate from their ideal values (i.e.

values that should be obtained under a given input as per the calibration). In addition, despite significant advances in technology, many sensors still remain vulnerable to factors like drastic changes in the operating environment, loose connectors, excessive temperatures etc. Another common source of variation in sensor readings is electrical noise that is picked up during transmission from the sensor to the data acquisition system or as a result of close proximity to components like electric prime movers, switching power supplies, etc. The effect of such noise is observed in the form of random, high frequency fluctuations in the sensor output signal. The resultant deviation in sensor readings can be classified into one of the following categories [Brignell and White, 1996]

- *Bias or Offset*: Bias is the difference between the theoretical and the actual sensor output value under the specified operating conditions. It can be considered as a constant value that gets added to the output value at every instant. Offset is the sensor output that exists when it should actually be zero.

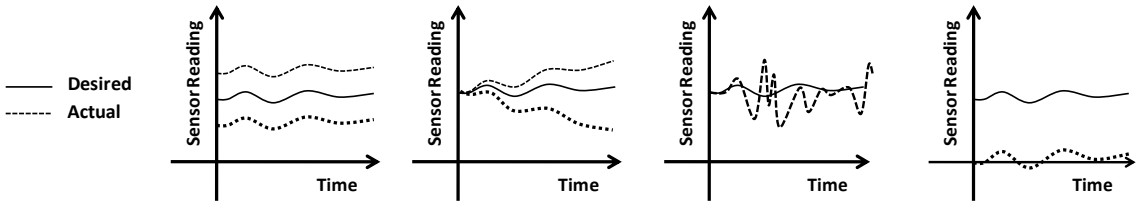


Figure 5-1: Types of Sensor Faults

- *Drift*: Drift is the gradual increase or decrease in the sensor readings (compared to its theoretical output) over an extended period. Drift usually occurs due to slow changes in the sensing element itself due to factors like wear, aging, improper installation, etc.
- *Cross Sensitivity*: This is the result of the sensing element responding to some physical phenomenon in addition to its primary measurand. The most common form of this behavior is with temperature. Other types of cross-sensitivities may also exist. However, this is less of an issue with state-of-the-art sensors where

careful design ensures that only the response to the desired measurand is maximized.

- *Failure*: A failure can be further classified into gradual, abrupt/complete or intermittent failure. Gradual failure may be seen in the form of increasing drift or bias in the readings. An abrupt failure is an obvious fault mode where there is no change in the sensor output in accordance with the input stimuli or there is no output from the sensor even in the presence of an actual input. This may be the result of a sudden shock impact on the sensor or a spike in the sensor supply voltage either of which may damage its internal components permanently. A failed connector may also be the cause of abrupt loss of sensor data even though the sensor itself may not be faulty. Intermittent failure is often referred to as 'precision degradation' in the literature. The effect of this is observed in the form of excessive noise in the sensor readings in certain operating regimes.

5.3. FAULT DETECTION AND ISOLATION

A sensor may be potentially faulty[2] if its output exhibits a behavior similar to those shown in Figure 5-1. The simplest way to ensure no sensor faults occur in a laboratory set-up is by regular recalibration. But this is often not feasible in systems like an EMA once it is deployed in an application environment. Another approach would be one of fault prevention through over-design. In the simplest form, this takes the form of physical hardware redundancy (using multiple sensors of the same kind) and the use of a majority voting scheme to rule out flawed sensors and to use the data from the ones that are not faulty (assuming that all the sensors do not develop faults at the same time).

Again, this is typically not a favored option in many cases due to factors like increased system complexity, size limitations, increased cost, etc. If multiple sensors cannot be used for each measurand, then the next best option could be to make each sensor intelligent enough to monitor its own characteristics and provide good data at all times. The SEVA (Self-Validation) concept proposed by [Henry, et. al. 1993] is based on

[2] The term 'sensor fault' as used in the context of this work is defined in detail in section 5.3.4.

the idea that, given sufficient on-board processing capabilities, a sensor can be made intelligent enough to provide more useful information than a simple measured value. SEVA devices are stand-alone in the sense that they monitor their own performance, validate the quality of data and provide a standard, formatted quality index for every measurement value. Such a device can provide enormous flexibility. With such sensors, the algorithms at the system level become much simpler since the liability of the data has already been assessed and characterized in a standardized fashion. It can also help in early detection and accommodation of recoverable sensor failures (with a reduced potential for permanent damage). However, such devices are far and few between and not very widely available. Thus, it is clear that there is a need for an analytical method to detect the occurrence and identify the source of faults in sensor data.

5.3.1. Issues in Fault Detection and Isolation Procedures

The underlying assumption in many approaches used for sensor fault detection is that the monitored system is operating normally and the unusual sensor readings are purely due to a sensor fault e.g. [Ibarguengoytia, 2003]. However, there may be cases where the fault doesn't lie in the sensor and its atypical behavior is caused by unforeseen changes in the monitored system (or any of its components). For instance, a fault in a motor winding increases its resistance which may be reflected in the form of a drop in the current drawn for a given supply voltage ($V=IR$). This situation would be reflected as a drop in the output of the current sensor which may be misinterpreted as a bias even though the sensor is operating correctly.

On the other hand, condition based maintenance algorithms that are used at the system level e.g. [Hvass and Tesar, 2004], while estimating the existing condition of the system, make the implicit assumption that all the sensors that are monitoring the system are operating correctly[3]. In such cases, using data from sensors with faults can result in incorrect estimates of the monitored system's capabilities and cause false alarms with regards to its estimated health or remaining useful life. In the worst case scenario, a

[3] Many CBM algorithms try to accommodate the presence of noise in sensor readings, typically as a zero mean, normally distributed value that is added to the actual reading

sensor as well as the system it is monitoring may be developing incipient faults and it may be impossible to distinguish between the two. Finally, a change in sensor readings might be simply due to a regular change in the operating conditions of the system e.g. change in the output of the speed sensor, when a motor controller ramps up the motor speed from standstill to its rated speed. But abnormal sensor behavior may sometimes be masked by such subtle changes in operating conditions, especially for anomalies like drift. The conundrum for any analytical procedure that is used to identify and mitigate faulty sensor data is thus to distinguish between these different scenarios and identify with some level of confidence the precise source of abnormality in the sensor readings when they occur.

5.3.2. Steps in a Fault Detection and Isolation Procedure

[Aradhye and Heger, 2002] define the phrase 'sensor fault detection, isolation and accommodation' to encompass the different facets of fault detection in sensors and summarize the requirements of the procedure as follows:

- Identifying a fault in the sensor/s within the shortest period
- Isolating the faulty sensor/s from the group of sensors in a system
- Classifying the type of fault affecting the faulty sensor/s
- Providing alternative values or estimates for the parameter under consideration, whose associated sensor is deemed to have failed

The steps mentioned above are also collectively referred to as 'sensor validation' by many researchers. From the perspective of the present work, the prime objective is to essentially determine whether or not the available sensor information can be used to update the stored performance maps. If it can be determined with some certainty that a deviation in a particular sensor's reading is due to a fault in the sensor, then data from that sensor will not be used to update a corresponding map. On the other hand, if it can be determined that a deviation in readings is due to a fault in the system, this would serve as additional knowledge that can be used by the higher level CBM algorithms (e.g. to increase the accuracy in localizing the cause of a fault to a particular system component)

and the sensor readings can still be used to update the stored maps to accurately reflect the extant system status. While the classification of the type of sensor fault may be important in some applications, it is considered of secondary importance in the present work and is not addressed here[4]. The framework adopted for fault detection and isolation is also capable of providing alternate estimates for physical parameters whose sensors are identified as being faulty.

Identification of a potential fault in the sensor readings is usually done by monitoring the variable of interest, generating residuals between its theoretical value (i.e. the value that should ideally be indicated by the sensor under normal conditions) and the observed values (the value that is actually indicated by the sensor) and monitoring these residuals with respect to pre-defined thresholds that represent the acceptable range of values for these residuals e.g. [Abdelghani and Friswell, 2007]. In other cases, residuals are not calculated and other methods like comparing means or hypothesis testing are used [Ibarguengoytia, 2003] [O'Reilly, 1998]. The estimation of the ideal value may be done in many ways based on the application- using look-up tables relating all the parameters of interest, tracking the measurand of interest over different time windows and using estimation algorithms like Kalman filtering (for sensors whose outputs can be modeled accurately) to create a validation gate or the region within which the sensor reading is expected to lie e.g. [Alag, et. al, 2001], or when multiple sensors are present in a system, by correlating the different measurands at various sampling instants, etc.

In each case, the approach used is one of analytical redundancy via inferential sensors i.e. using sensors other than the primary sensor to infer the measurand under consideration [Xu and Kwan, 2003]. In its most intuitive form, such an approach requires the development of comprehensive mathematical models representing the relationships between the monitored parameters, which is not always feasible (due to the highly nonlinear and coupled nature of the physical variables involved) and in some cases may not be even possible. With multiple and diverse sensors in the system, the number of

[4] Although it is possible to extend the Bayesian network based fault detection and isolation framework presented in Section 5.4 to do this.

relations that are needed to correlate them and that are needed to be evaluated simultaneously also increases[5]. In addition, the models are susceptible to errors (e.g. errors that may be introduced as a result of simplifying assumptions made during the modeling process) and may provide incorrect estimates of the parameter of interest. This may result in missed faults or false alarms. These models are often developed for a system under a specified set of conditions. Any change in the system characteristics may entail significant changes to be made to these models, which requires considerable time and expertise.

5.3.3. Review of Existing Methods in the Use of Bayesian Belief Networks for Fault Detection and Isolation

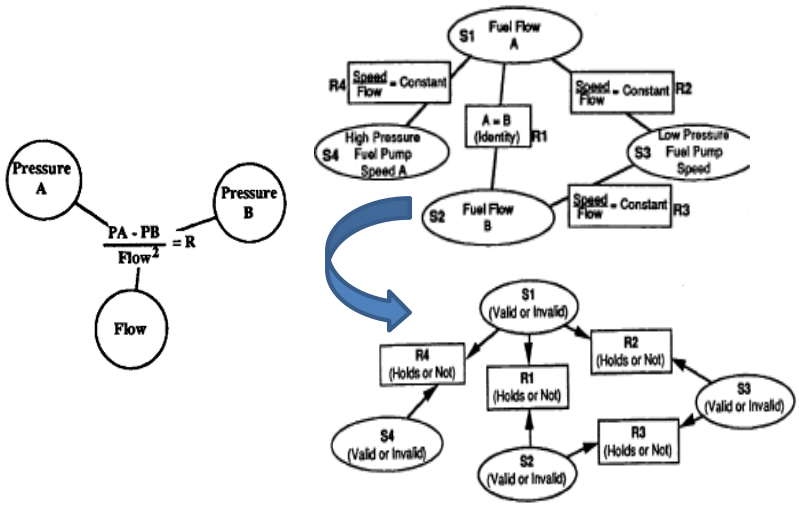
Different researchers have explored a diverse range of techniques to approach the problem of fault detection and isolation using analytical redundancy. These include fuzzy logic [Frolik, et. al. 2001], using the Nadaraya Watson statistical estimator [Wellington, et.al, 2002], Kalman filtering [Alag, et. al, 2001], using Principal Component Analysis [Dunia and Qin, 1997], using subspace model identification [Jiang, et. al., 2007]. Although each method has its own advantages (speed, accuracy, ease of implementation, etc.) and weaknesses (need for a system of mathematical models/equations, inability to detect multiple sensor faults, inability to distinguish between sensor and system faults, need to integrate different approaches together in the same application in order to accomplish different tasks like modeling, fault detection, fault isolation, etc.), the focus of this work is to use the Bayesian causal network framework developed in earlier chapters to accomplish these goals. This approach can provide a unified, data-driven framework for correlating the system variables in a physically meaningful manner (that can also be represented graphically for intuitive understanding) as well as perform fault detection, isolation and fault accommodation using the same framework. In addition, the existence of a well developed mathematical formalism based on probability theory, helps account for the nonlinearities and uncertainties associated with the system under

[5] Even with the computing resources presently available, evaluating these models to produce estimates of the measurand of interest may be a computationally challenging task

consideration. The use of Bayesian networks in conjunction with or to augment other techniques for fault detection has been explored by researchers [Castanon, et. al., 2008], [Verron, S., et. al., 2008]. But the use of Bayesian causal networks seems to be more widespread in the area of fault diagnosis (with fault sources modeled explicitly) than in fault detection e.g. [Liu and Zhang, 2002], [Riascos, et. al, 2006], [Yongli et.al, 2006], [Mast, et.al, 1999]. The following section provides a brief overview of the use of Bayesian belief networks by researchers in various domains to address the issue of sensor fault detection and isolation.

5.3.3.1. Approach I

[Bickmore, 1994] combined empirically derived mathematical relations with Bayesian belief networks to provide real-time validation capability for multiple channels of sensor data. The goal was to develop software to detect sensor failures on rocket engines in real time and avoid the engine controller from erroneously shutting down the engine by misinterpreting sensor failures as engine failures. The author presents an approach based on converting an undirected validation network, consisting of a set of sensors and the analytical relations among them, into a bipartite Bayesian belief network for validation.



(a) **(b)**

[Bickmore, 1994]

Figure 5-2: Bayesian Network for Sensor Validation in Rocket Engines

Figure 5-2 (a), for instance, shows the relation between pressure and flow sensors using the standard formula for fluid line resistance. The relations involving each sensor are evaluated and by thresholding on the residuals (between the calculated and the actual measured values) based on normal engine firing test data, the probabilities of the relations holding true are calculated. A sensor is then flagged as faulty or operating correctly based on the number of relations that involve the sensor under consideration and that hold true concurrently.

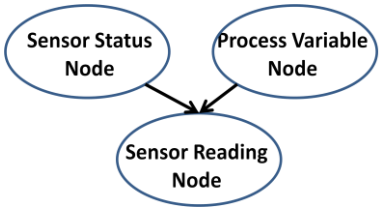
The main characteristic of this approach is that it relies on the availability and accuracy of the analytical relations which are derived based on initial testing data which only represent the normal engine operating conditions. The results presented in this work demonstrated that successful distinction between a sensor and plant anomaly could be achieved only if all the relations held true and no single engine anomaly rendered all the relations involving a particular sensor to be invalid.

5.3.3.2. Approach II

[Aradhye, 1997] demonstrated the use of Bayesian networks (with both discrete and continuous-valued nodes for systems under steady state and using dynamic Bayesian networks, again with discrete and continuous-valued nodes, in conjunction with different schemes like observers, neural networks, etc. for dynamic systems respectively) for sensor fault detection, isolation and accommodation using the basic model as shown in Figure 5-3. The three nodes represent the sensor status node (which indicates whether the sensor is faulty or operational), the process variable node (representing the theoretical or expected value of the measurand) and the measured sensor reading node (which represents the actual value of the measurand indicated by the sensor). The sensor readings available at any instant are set as evidence in the network and using standard propagation techniques, the probabilities of the two parent nodes are updated. If the objective is only to perform fault detection, two states {*faulty*, *operational*} are suggested for the sensor status node. To classify the specific type of fault occurring in the sensor, the use of four states for the sensor status node {*normal*, *bias*, *precision degradation* and *complete*

failure} was suggested. The distinctive choice of prior probabilities for each of these states was based on their mutually exclusive effects on the probability distribution of process variable node. As in the earlier case, the sensor readings available at any instant are instantiated as evidence and the updated probabilities of the various states of the sensor status node are used to identify and classify faults. The main feature of this approach is that the one-to-one mapping between the different modes of operation like normal, bias, etc. may not be always possible [Aradhye, 1997].

In addition, the use of continuous valued nodes required the linearization of the process model and the assumption of Gaussian distributions for all the continuous variables (both of which are not needed in case of discrete valued nodes which are purely based on available data). The use of observer and neural networks augmented schemes were also subject to errors arising from the accuracy of the observer, estimates of partial derivatives in the neural network, etc. as described in [Aradhye, 1997], thus necessitating the need for the parallel existence of multiple such schemes. For simultaneous detection of sensor and process faults, [Aradhye, 1997] suggests the use of additional nodes to model those faults. Since this means explicitly representing the fault as part of the network, it requires knowledge of the actual faults that affect the process and the effect of those faults on the process variables (reflected as a change in the conditional probability distributions of those variables). For instance, [Aradhye, 1997] uses a known analytical relation to model a leak fault that changes the relation between input and output temperatures in a chemical stirring process. However, such relations may not be always easily available to model the faults apriori. Despite these issues, this work presented a unified framework to perform sensor



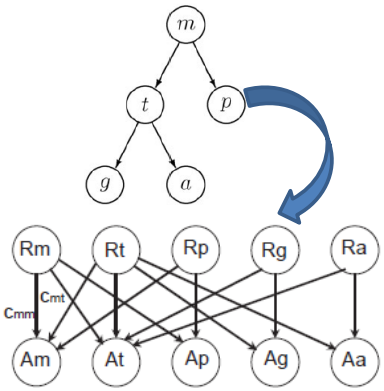
[Aradhye, 1997]

Figure 5-3: Basic Bayesian Belief Network Unit for Sensor Fault Detection, Identification and Isolation

fault detection and classification as well as detecting a fault in the monitored system using Bayesian networks.

5.3.3.3.Approach III

[Ibarguengoytia, et. al, 2005] identify and isolate the faulty sensors using two separate Bayesian networks. To identify a faulty sensor, the authors used the concept of an Extended Markov Blanket (*EMB*) i.e. the set of nodes consisting of the sensor being checked for fault and all the other sensors in its Markov Blanket (or *MB*- the set of nodes consisting of all the direct parents of a node, all the direct children of the node and all the spouses of the node i.e. the other direct parents of the direct children of the node). Each individual sensor is validated by instantiating the other sensors in its *MB* with the measured values indicated by those sensors. The order in which the sensors are evaluated is based on the contribution of each sensor to the overall entropy (in an information theory sense) of the system. By using two fault detection criteria (*p* value and mean) to compare measured and estimated values of the variables obtained by probabilistic propagation, a list of potentially faulty sensors is obtained. This set of potentially faulty sensors is then compared with a table of *EMB*s for each sensor and based on different levels of matching, one or more sensors may be identified as being faulty.



[Ibarguengoytia, et. al, 2005]

Figure 5-4 : Use of Bayesian Networks for Sensor Validation in Gas Turbines

The authors also outline the conditions, wherein multiple faults can be diagnosed or where the faults may be indistinguishable. Since the algorithm yields a list of potentially faulty variables only after a complete cycle of validation, to facilitate any time behavior, the authors propose the use of a second Bayesian network based on a canonical

noisy-*OR* model [Pearl, 1988] and consisting of two levels of nodes- a root node level representing each individual variable and the real faults in them and a level representing apparent faults in the sensors (i.e. a sensor which may have been incorrectly deemed as faulty because of the existence of a real fault in one or more of the other sensors that were used to estimate its value during the fault detection process) with arcs from every root node to all the nodes in its *EMB* (Figure 5-4). For any-time validation, the authors develop and pre-compile a binary decision tree, where the links are again based on the variable that gives the best improvement in the average entropy of the system. The final result of validation is also validated using an entropy-based function. The authors demonstrate the effectiveness of their method by using it to validate a set of 21 temperature sensors in the start-up phase of a gas turbine.

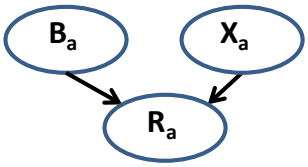
The main features of the approach presented in [Ibarguengoytia, et. al, 2005] are its ease of implementation, the ability to detect multiple sensor faults under some conditions, and its any-time capability. However, the procedure assumes that any deviation in the sensor readings is exclusively due to a sensor fault and there are no changes in the monitored system itself.

5.3.3.4. Approach IV

[Mehranbod, et. al. 2003] present the use of Bayesian belief networks (BBN) to perform sensor fault detection as well as identification. The authors propose a new single sensor BBN model similar to the model proposed by [Aradhye, 1997] and use it as a building block to develop a complete model for multiple sensors. The functional mathematical model used is $R_a = X_a + B_a$ where R_a represents the sensor reading, B_a represents the bias and X_a is the actual value of the measurand (Figure 5-5). For the nodes X_a and R_a the use of five states is suggested {very low (L_2), low (L_1), normal (N), high (H_1) and very high (H_2)}. For the B_a node, again the use of five states {very negative (n_2), negative (n_1), zero (z), positive (p_1), very positive (p_2)]} is suggested. The distinguishing characteristic of this model is the number of states used for the B_a node (corresponding to the sensor status node of [Aradhye, 1997]) and its use to identify the type of fault, which

presents a more realistic representation of the variation in sensor readings as compared to the one-to-one mapping between the different modes of operation (normal, bias, etc) as presented in [Aradhye, 1997]. The model design characteristics like bin sizes for the different types of nodes, prior and conditional probability data, etc. are derived systematically taking into account the acceptable range of deviations in the sensor readings, the sensor accuracy, etc. by using an optimization process and Monte Carlo simulations.

The authors define the concept of Probability Absolute Difference (PAD) i.e. the difference between the updated probability distribution of the B_a node, and its distribution under known, normal operating conditions to define a novel threshold setting procedure that can be used to signal faults for all the sensors in the system. The *PAD* and the sum of *PAD* (or *SPAD*) are used to define a normalized fault detection index D_i to identify a faulty sensor ([Mehranbod, et. al. 2003] emphasize threshold setting for detection of faults based on this value to avoid false alarms and missed faults) and a fault identification index I_i to help classify the exact type of fault (drift, bias, etc.). [Mehranbod, et. al. 2005] also introduce the concept of adaptable nodes to extend the framework discussed above to perform fault detection and identification in a transient process. In both cases, the authors demonstrate the capabilities of the model to detect and identify different types of sensor faults such as bias, drift and precision degradation in sensor readings for both single and simultaneous multiple faults by considering several scenarios involving the behavior of different sensors integrated in applications like polymerization reactors and liquid-level systems. However, the main characteristic of this work as in the earlier approaches is the underlying assumption that there are no shifts in the system operating characteristics.

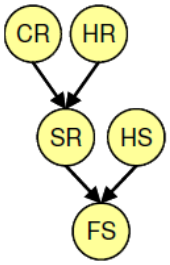


[Mehranbod, et. al. 2003]

Figure 5-5: Single Sensor Bayesian Belief Network Model for Sensor Fault Detection and Identification

5.3.3.5.Approach V

[Mengshoel, et. al., 2008] investigate the use of Bayesian networks to perform sensor validation in spacecraft electrical power systems using the Advanced Diagnostics and Prognostics Testbed (ADAPT) at the NASA Ames Research Center as a case study. The Bayesian network model (Figure 5-6) consists of discrete nodes to represent the different components in the system like relays in an electrical power system (*CR* and *SR* represent the command to the relay and the relay status) and sensors used to monitor them (*FS* represents the relay's feedback sensor) with the addition of explicit nodes (*HS* and *HR*) to represent the health status of the sensors as well as the other components (e.g. relays). The method used is to dynamically provide inputs to the network using the sensor readings and commands to the relays and posing a joint probabilistic query (e.g. *MAP*- Maximum A Posteriori) instead of analyzing marginal distributions of the individual nodes separately for validation.



[Mengshoel, et. al. 2008]

Figure 5-6: Bayesian Network for an Electrical Power System

By choosing the appropriate evidence nodes and query nodes (in this case, the health nodes for the different sensors and relays), the authors demonstrate that it is possible to isolate and distinguish between component faults and sensor faults. The use of reasoning about the joint probability of all the nodes in the network also allows diagnosis of multiple sensor faults. The Bayesian network is compiled into an arithmetic circuit that enables fast inferencing for the purpose of real-time operation. [Ricks and Mengshoel, 2009] extend the work described above which included only abrupt discrete faults, to include cases with abrupt continuous (or parametric) faults such as continuous stuck faults and continuous offset faults, using explicit fault nodes.

Although both [Mengshoel, et. al. 2008] and [Ricks and Mengshoel, 2009] demonstrate the effectiveness of this approach by testing on networks with hundreds of

sensor and component nodes, the use of explicit health nodes (with a specific number of normal and abnormal states) to modify the behavior of the nodes representing the sensor and the component health nodes is a potential drawback. It assumes a finite number of operational fault modes and that data regarding the behavior of the system components and sensors in their faulty conditions is available. However, such information is often hard to come by or even difficult to estimate since typically all the potential fault modes may not be known in advance.

For instance, consider a current carrying conductor with a specific voltage applied across it and being monitored by a current sensor. A fault in the internal sensing element or a change in the conductor resistance may both cause a change in the reading from a current sensor but a drop in the supply voltage level will also affect the reading, albeit differently. Different failure modes may also have different effects; for instance, a bias fault may affect a sensor reading differently from a drift fault.

Table 5-1: Literature Review of Research Using Bayesian Networks for Sensor Fault Detection and Isolation

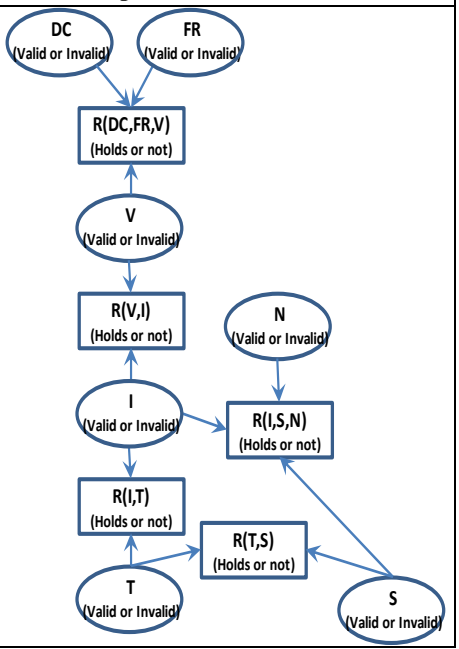
Method	Features	Representative Network
[Bickmore, 1994]	**Potential advantages**: • Realtime operation • Ability to handle large number of sensors • Provides analytical redundancy, well-defined error rates (based on MTBF, cycle time, standard deviation, etc.) **Potential disadvantages**: • Requires invariant analytical relations (derived from empirical data obtained under normal operating conditions) between all variables of interest • Cannot robustly handle multiple sensor failures • Distinction between sensor/system faults possible only with multi-point sensor failure	

Table 5-1: Literature Review of Research Using Bayesian Networks for Sensor Fault Detection and Isolation (continued)

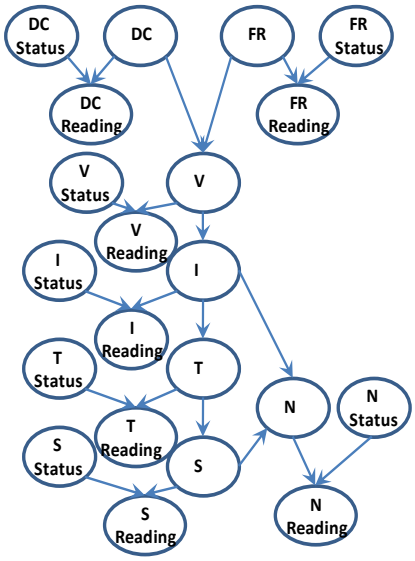
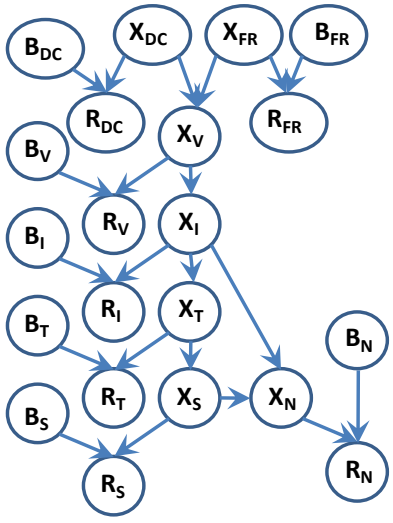
Method	Features	Representative Network
[Aradhye, 1997]	**Potential advantages**: • Ease of implementation (demonstrated for both continuous and discrete data, in steady state and under transient conditions) • Unified framework for fault detection (sensor as well as system) as well as classification (normal, bias, precision degradation, complete failure) • Allows incorporation of system specific qualitative knowledge **Potential disadvantages**: • Explicit modeling of faults (with known effects on variables of interest) required to detect as well as distinguish between sensor and system faults • One-to-one mapping of modes of sensor operation for fault classification not realistic • Potential for false alarms due to rigid definition of fault modes	
[Mehranbod, et. al. 2003]	**Potential advantages:** • Ease of implementation, demonstrated for both steady state and transient conditions • Novel threshold setting procedure, fau detection and identification index t enable effective fault identification a well as classification (bias, drift precision degradation) • Ability to handle multiple sensor faults **Potential disadvantages:** • Assumes no variation in system condition (cannot detect system faults) • May result in large CPTs for variables that have a wide operating range • Thresholds determined for variables with spread out distributions (i.e. low probability values for different states) may be unreliable.	

Table 5-1: Literature Review of Research Using Bayesian Networks for Sensor Fault Detection and Isolation (continued)

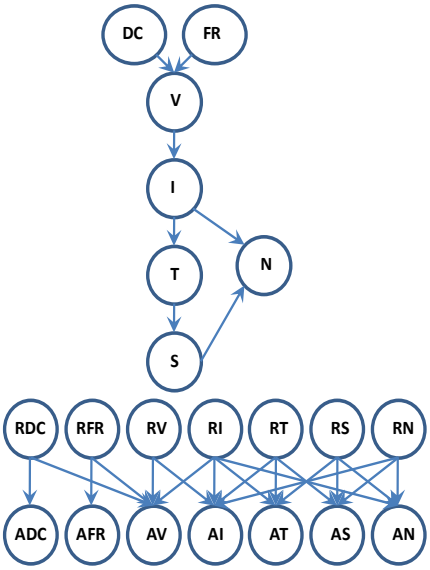
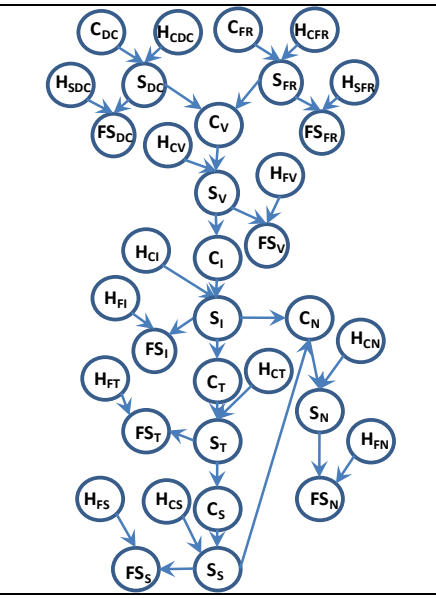
Method	Features	Representative Network
[Ibarguengoytia, et. al., 2003]	**Potential advantages:** • Anytime algorithm (in conjunction with a pre-compiled binary decision tree) • Ability to detect multiple sensor faults under specific conditions • Ability to isolate faults to a specific cause (using quality measures based on information theory) **Potential disadvantages:** • Assumes no variation in system condition (cannot detect system faults) • Requires two separate networks • Model learned from data for specific operating conditions and may not be applicable over the entire operating regime of the system	
[Mengshoel, et. al., 2008]	**Potential advantages:** • Real-time operation (fast inferencing by compiling the Bayesian network into arithmetic circuits) • Proven for very large networks (hundreds of nodes) • Enables distinction between sensor and system faults **Potential disadvantages:** • Explicit modeling of faults required (requires apriori knowledge of all fault modes and their effect on associated variables) • Intricate network structures are needed even for moderately complex systems	

Table 5-1 shows a comparison of the different approaches described thus far. Each row in the table contains an illustrative network, representing the structure equivalent to the simplified network discussed in Section 5.6 that would result by adopting the approach specified. To address some of the shortcomings, the following

section presents a new approach to sensor fault detection and isolation using Bayesian networks that can take into account single or multiple sensor failures as well as deviations in sensor readings that may be caused due to issues with the monitored system.

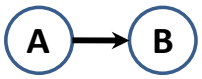
5.3.4. Definition of Sensor and Process Faults

With multiple sensors in the system, if a single sensor indicates values that seem to be out of line whereas the rest of the sensors indicate their expected values, it is likely that there could be a potential issue with that particular sensor. Hence, the connotation of the term '*sensor fault*' as used in the context of the present work only indicates the disagreement between the ideal or theoretical value that the sensor is supposed to indicate under the prevalent operating conditions and the measured value it actually indicates at the sampling instant under consideration and does not necessarily mean that the sensor itself is flawed. This difference may be caused by a temporary drift, bias or noise in the reading. Hence, the output from the particular sensor would need to be tracked over multiple sampling instants to declare with certainty that the sensor itself is faulty. Such tracking may also help in classifying the specific type of sensor fault e.g. as described in [Mehranbod, et. al., 2003].

If however, at the given sampling instant, the readings for most of the sensors in the system seem to be deviating from their expected values, then it might be indicative of a potentially more serious problem than faulty sensors (the logic here being that the sensors are by and large very reliable and it is an unusual event when multiple sensors are failing simultaneously). Such changes might be a result of incipient faults in one of the system components. As seen earlier, the different nodes in a Bayesian network represent the physical variables (e.g. torque, speed, etc.) pertinent to the system and its components. Thus, the link between every pair of nodes can be considered to be a '*process*' that converts the physical parameter represented by the parent node into the parameter represented by the child node. For discrete variables, the strength of the correlation between the variables represented by a parent node-child node pair can be

said to be quantified by the conditional probability table (CPT) of the child node [Korb and Nicholson, 2004].

Consider a Bayesian network (Figure 5-7) consisting of two nodes *A* and *B* corresponding to some physical variables measurable by using appropriate sensors. For instance, *A* can represent the voltage applied across a wire and *B* may represent the current flowing through it. The link *A->B* between the two nodes indicates that they are causally related[6] i.e. $B= f(A)$ and thus represents the '*process A->B'*. The parameters or the entries in the CPT of *B* represents the conditional probability distribution $P(B=b_i|A=a_j)$, where $i=1...m$ and $j=1...n$, are the number of states of *B* and *A* (the states represent the distinct values that these two variables can assume) respectively. If there is simply a change in the operating conditions of the system, the conditional distribution for the child node given the value of its parents would still hold valid but a fault in any system component would essentially render the relation between *A* and *B* as encoded in the CPT of *B* invalid. In other words, this would be reflected as a change in the parameters of the CPT of the child node *B*. Hence, the term '*process fault*' in the context of this work can be defined as any such change in the relations between pairs of variables represented in the network.



Original CPT for B

	$A=a_1$	$A=a_2$
$B=b_1$	0.9	0.2
$B=b_2$	0.1	0.8

Modified CPT for B

	$A=a_1$	$A=a_2$
$B=b_1$	0.2	0.7
$B=b_2$	0.8	0.3

Figure 5-7: Example Bayesian Network with Two Nodes

Figure 5-7 shows two states each for *A* and *B* (the values are illustrative). The subscripts denote different operating conditions. Under normal circumstances, for the condition $A=a_1$ it can be seen from the CPT, that a value of b_1 is most likely to be obtained for *B*. If the operating condition changes to $A=a_2$, then the corresponding most

[6] This does not necessarily mean that a closed form analytical equation must exist that between *A* and *B*, although there may be one.

likely value for B is b_2. The A->B process can be said to be perfect or non-faulty under these conditions. If however, the relation between A and B changes due to some fault in the system components, as shown in the modified CPT for B, for the operating condition $A=a_1$, a value of $B=b_2$ is most likely to be obtained as per the modified relation. This will be construed as a process fault in the A->B link.

The Bayesian network representation of any complex system would include a number of processes representing the relation between the physical variables relevant to its different components. A fault in one or more of these components will be manifested as a change in the relations between the variables associated with it. Thus identification of a process fault can be indicative of a potential system fault. For instance, a reduced current reading may be because of a decrease in the external load acting on the motor (requiring the control system to generate a lower torque to maintain a specific command speed) or it may be the result of a winding fault thereby causing an increase in the motor phase resistance, thus decreasing the current I drawn for a fixed supply voltage V. The first case would simply be considered a change in the operating conditions. The latter case can be considered a process fault occurring in the V->I process which occurs in the motor windings, thereby indicating a potential problem in the windings.

5.4. SENSOR AND PROCESS FAULT DETECTION AND ISOLATION (SPFDI) ALGORITHM

The stored performance maps are periodically updated only after it is ensured that the system and all its sensors are perfectly operational[7]. Thus, the last updated set of maps (or for a new system, the certification maps), represent the latest known information available to the operator regarding the system status before information from a new set of sensor readings becomes available. The CPT for every node in the network (between consecutive updates) contains data derived from these maps. Any value deduced using the network represents the value that should ideally be obtained from the sensor corresponding to that node if there are no new or unknown problems in either the

[7] Any faults or change in the system condition until that point have been assumed to be accounted for and the system is deemed worthy of continued performance to accomplish the task at hand (via CBM).

sensors or in any of the system components that have not already been accounted for. On the other hand, the sensors are sampled at a much faster rate compared to the pace at which maps are updated. The values indicated by the sensors at any sampling instant indicate the extant status of the system at that instant and will be therefore be influenced by any possible issues that have occurred since the maps were last updated.

The premise of the proposed algorithm is that by sequentially instantiating different nodes in the Bayesian network (referred to as '*Nodes Instantiated*' or *NI*), performing probabilistic propagation, and examining the resultant values of other specific nodes in the network (referred to as '*Node of Interest*' or *NoI*), it may be possible to estimate the validity of the sensor readings obtained. The readings from the sensors corresponding to *NI* are used to set specific states of such nodes as evidence to the network. Every node in the network can be considered as a *NoI* sequentially one at a time, until all the nodes in the network have been traversed. For each *NoI*, multiple values can be estimated by considering different combinations of other nodes in the network as *NI* and calculating its posterior probability distribution. The values inferred for the *NoI* are compared with the actual values indicated by their corresponding sensors to determine if the sensors are indicating what they are supposed to under the prevalent system operating conditions.

For each of the many *NI* and *NoI*, if the values indicated by the sensors and the inferred values concur (for the *NoI*), then it indicates that the system has not changed significantly from its last known condition. Hence, the assurance in the fact that the sensors are operating normally and the CPT parameters for each node in the network continue to maintain the same values as before (i.e. the presumed relations between the different physical variables, referred to as '*processes of interest*' or *PoI* henceforth, remains the same) increases. Conversely if these values do not match, the assurance decreases. By assigning a numerical measure to this level of assurance in the different sensors and different links in the network, and incrementing or decrementing the measure suitably each time the measured and estimated values are compared for different nodes of interest, it is possible to estimate the cause of undesirable deviations in the sensor

readings when they occur. This can help identify potential sensor faults. In addition, it can also help verify if the presumed conditional relations between the different variables in the network, encoded in the CPTs of all the non-root nodes still hold true[8] to help identify system faults.

5.4.1. Sensor and Process Fault Detection and Isolation- A Thought Experiment

Consider a network with two nodes of the form $A->B$. Assume that the probability distributions for both the nodes are appropriately discretized into the required number of states of the form $A= a_i$ and $B=b_j$, where $i=1...p$ and $j=1...q$; the bin sizes Δa and Δb are determined taking into account the sensor accuracy and the application requirements[9]. The values indicated by the sensors for A and B would then be taken to correspond to one of their states a_i and b_j respectively. Suppose the readings a_m and b_n are indicated by the two sensors respectively.

The value of the random variable represented by node B can be determined using the network, by setting the measured value $A=a_m$ as evidence (*NI*) and performing probabilistic propagation[10]. The state of B (*NoI*) with the highest probability should ideally be the same as b_n (only the posterior marginal distribution of the *NoI* is considered) if the system has not changed since its parameters were last updated. The value $B=b_n$ will be obtained after propagation if and only if the correct value of A was set as evidence (i.e. one which corresponds to the appropriate column in the CPT of B) and the relation between A and B has not changed (i.e. the values in the CPT for B are still valid). If that is indeed the case, then it can be said that the assurance in the fact that the sensor for A is operating correctly is now increased since it produced the desired value for B. Similarly, the assurance in the fact that the $A-> B$ process still maintains the presumed relation between A and B is also now increased since the desired value for B was obtained as per the conditional distribution presently available in its CPT. Finally,

---

[8] The structure of the model remains unchanged throughout the process

[9] [Mehranbod, et. al, 2003] provides a detailed description of this process for a chemical engineering application but their suggested methodology is generally applicable to any domain.

[10] Probabilistic propagation generates a distribution for the different values of B based on its CPT. The term 'value' as used here refers to the state of node B with the highest probability of occurrence.

assurance in the fact that the sensor for B is operating normally is also increased since it indicates a value that was expected. If however, by setting a value of $A = a_m$ as evidence to the network and after propagation, the value obtained is $B = b_o \neq b_n$, then it could mean one of three possibilities:

a) A wrong value was set as evidence to the network. This would imply that the sensor for A is potentially faulty as the reading for A is incorrect (as $A=a_o \neq a_m$).

b) The relation $A->B$ has changed and the CPT for B no longer accurately represents its conditional dependency with respect to A. For a given value of $A = a_m$, a value of $B=b$ cannot be obtained because the parameters of the conditional distribution have changed due to some change in the system or it components.

c) The sensor for B is faulty, resulting in an incorrect value ($B=b_o \neq b_n$), even though the correct value was set as evidence and the conditional distribution is still valid.

In the situations described above, the assurance in the fact that the sensor for A is operating correctly, the process $A->B$ maintains the original relationship, and the sensor for B is operating correctly can be said to have decreased respectively because of the discrepancies in the expected and observed values for the sensors. In this particular example, with two sensors and one process, it is not possible to distinguish between a sensor fault and a process fault since there is no additional information, using which, one can estimate and cross-examine the values for A and B, it is not possible to isolate which of the three factors mentioned above are responsible for the mismatch in values.

5.4.2. Steps in the SPFDI Algorithm

The SPFDI method described above can now be extended to a more general Bayesian network with complex structures or a larger number of nodes. The first step is to decide the types of instantiations that can be done considering different nodes in the network as *NI* and determine the *NoI* and *PoI* associated with each such instantiation. The second step is to quantify the level of assurance in the different *NoI* and *PoI* represented in the network using an appropriate measure, and modifying it based on the results of comparing the measured and the inferred values for a particular *NoI*. The intention is to

provide an intuitive metric to enable the operator to make a judgment regarding the status of different sensors and processes (i.e. whether they are potentially faulty or not) at the end of a fault detection and isolation procedure.

5.4.2.1.Generating the Instantiation Table

Consider again, a simple network *A->B*. With this structure, there are two possible types of reasoning possible "Given the value of *A* what is the value of *B*" or "Given the value of *B* what is the value of *A*". The former can be described as *deductive* or *predictive* reasoning [Korb and Nicholson, 2004] and represents a cause-to-effect reasoning where *A* is the evidence node and *B* is the query node. The latter is referred to as *abductive* or *diagnostic* reasoning [Korb and Nicholson, 2004] and represents an effect-to-cause reasoning where *B* is the evidence node and *A* is the query node.

As described in [Ashok and Tesar, 2007], the decision-making methodology in EMAs adopts a Bayesian causal network approach where the arrows depict causality as well as conditional independence (the network is a D-map and an I-map). Although mathematically the flow of information in the process of conditioning on the evidence nodes and updating the belief of the query nodes can happen in either direction, it is important to factor in the physical relevance of the variables represented in the network to decide the type of reasoning to adopt. For instance, in Figure 5-7, suppose the node *A* represents the total current drawn by the motor and the node *B* represents torque produced by the motor. It indicates that the domain expert, who designed the network, has made use of the knowledge that in a motor, the current drawn causes the resultant motor torque and not the other way around. Hence, in order to judge how good the process of conversion of current to torque is, a logical method would be to consider current as the input or the control variable which is manipulated in order to observe the variation in the response variable i.e. torque. Extending the argument to the larger network case, in the proposed algorithm, the *NI* are always chosen from the set of ancestral nodes for a given *NoI*. This provides an intuitive starting point to implement the proposed SPFDI algorithm.

Consider the network in Figure 5-8. The system represented by this network has four sensors corresponding to the physical variables represented by *A*, *B*, *C*, *D* and three processes *A->B*, *B->C*, and *C->D*. There are six possible instantiations that can be done ancestrally with this network. Note that, when the *NI* and the *NoI* are separated by a number of intermediate nodes, all the intermediate links are included as the processes of interest in that instantiation step. Considering node *D* as the *NoI*, the *NI* can be {*A, B, C*}, with the processes *A->B*, *B->C*, and *C->D* as the *PoI*. With *C* as the *NoI*, the *NI* can be {*A, B*}. The two processes *A->B* and *B->C* are the *PoI*. A comprehensive list of all such possible instantiations (considering different nodes/sensors in the network either as *NI* or *NoI*, and all the intermediate processes as *PoI*) can be represented as a table referred to as the '*instantiation table*' for that network (Table 5-2).

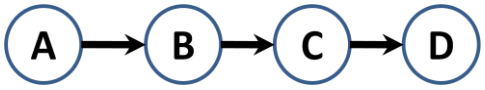


Figure 5-8: Example Bayesian Network

Table 5-2: Example of an Instantiation Table

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
1	*NI*	*NoI*			*PoI*		
2	*NI*		*NoI*		*PoI*	*PoI*	
3	*NI*			*NoI*	*PoI*	*PoI*	*PoI*
4		*NI*	*NoI*			*PoI*	
5		*NI*		*NoI*		*PoI*	*PoI*
6			*NI*	*NoI*			*PoI*

As will be seen later, the composition of the instantiation table plays an important role in determining the efficacy of the algorithm in isolating sensor and process faults. The process of probabilistic propagation considering each row of the instantiation table until all the rows have been exhausted is called a '*fault detection and isolation cycle*'[11].

[11] There are additional possibilities which include use of abductive reasoning by setting descendent nodes as *NI* and considering ancestral nodes as the *NoI*, or setting only the nodes in the Markov blanket of a *NoI* as the *NI* nodes, etc. These cases are not presently considered and may be pursued at a future stage.

5.4.2.2.Using the Instantiation Table

Suppose the readings indicated by the sensors for the nodes *A*, *B*, *C*, *D* are *A=a*, *B=b*, *C=c*, and *D=d* respectively. Let a_{inf}, b_{inf}, c_{inf}, d_{inf} be the corresponding values (node states) that are obtained via probabilistic inference using the network[12]. The first row indicates that node *A* is the *NI* and the *NoI* is *B*. The value (state) *A=a* is set as evidence and an inference is drawn regarding the most probable value of *B*. If it is observed that b_{inf} *=b,* then the assurance that the sensor for *A* is operating correctly, the sensor for *B* is operating correctly and the process *A-> B* still maintains the presumed relation between *A* and *B* are all increased since the desired value for *B* was obtained as per the available conditional distribution in its CPT. If, on the other hand, if the values do not match, it could be indicative of a potential fault in any of the three network components i.e. the sensors for *A*, *B* or the process *A->B*.

The second row in the table indicates node *A* as the *NI* and node *C* as the *NoI*. Thus the value *A=a* is again set as evidence but an inference is now drawn regarding the most probable value of *C*. If c_{inf} = *c*, then the assurance that the sensors for *A* and *C* are operating correctly, the intermediate processes *A->B* and *B->C* maintain the relationship as specified by the CPTs of *B* and *C* are all increased. The agreement in the measured and inferred values of *C* implies that the value of *A=a* resulted in a correct value of *B=b* after probabilistic propagation, which in turn resulted in the correct value of *C=c* after the next level of probabilistic propagation. If the values c_{inf} and *c* are different, then it can be said that the assurance in each of the network components mentioned above is reduced.

A similar logic may be used to interpret the remaining rows of the instantiation table. Row 1 provides a judgment regarding the status of the sensors for *A*, *B* and the process *A->B*; row 2 provides a judgment regarding the status of the sensors for *A*, *C* and the processes *A->B* and *B->C* and so on.

[12] When determining these values, the most probable values are considered, i.e. the state with the largest probability value in the posterior marginal distribution of nodes *A, B, C, D*.

5.4.2.3.Quantifying the Level of Assurance in Sensors and Processes

One way to quantify the level of assurance in the different *NoI* and *PoI* is to assign unique weights to each node (sensor) and process in the network, to indicate the belief the operator has regarding their status as being faulty or not. Let these belief values be W_S and W_P respectively; where S represents a sensor corresponding to a node and P represents a process in the network.

When a system and its sensors are newly deployed, the assurance that all the sensors and processes are operating correctly is quite high. But the same cannot be said for a system which has been operational for some time and its sensors have been exposed to the ambient and operating conditions. Hence, to provide an intuitive meaning to W_S and W_P, the range of values they can assume are defined to lie in the interval [0, 1]. The values tending to zero imply that the assurance in that element (i.e. node or process) of the network being 'healthy' is very low. Conversely, the values tending towards 1 imply perfect health.

A vital factor to be considered is the initial belief value assigned to each sensor or process. Precise sensor reliability information is often scarce as it is highly dependent on the conditions under which the sensor is used. Although most manufacturers do provide some guidelines on sensor life under specified conditions, it is prudent to judge the health of a sensor based on analyzing its output under the prevailing conditions. With respect to judging how good a process is, the situation is more complicated since the only sources of information about the process are the sensors that measure the constituent variables in the process. Hence, at the start of each new fault detection and isolation cycle, it is assumed that there is no knowledge of the status of sensors or processes.

To account for this ignorance regarding the initial conditions, the beliefs for all the nodes and processes is initialized to a value of 0.5. This implies the assumption of an equal likelihood of the particular component being faulty or perfectly operational[13].

[13] An analogy can be drawn here considering the simple example of tossing a fair coin where the probability of getting a heads or tails is equal $P(\text{heads})=P(\text{tails})=0.5$

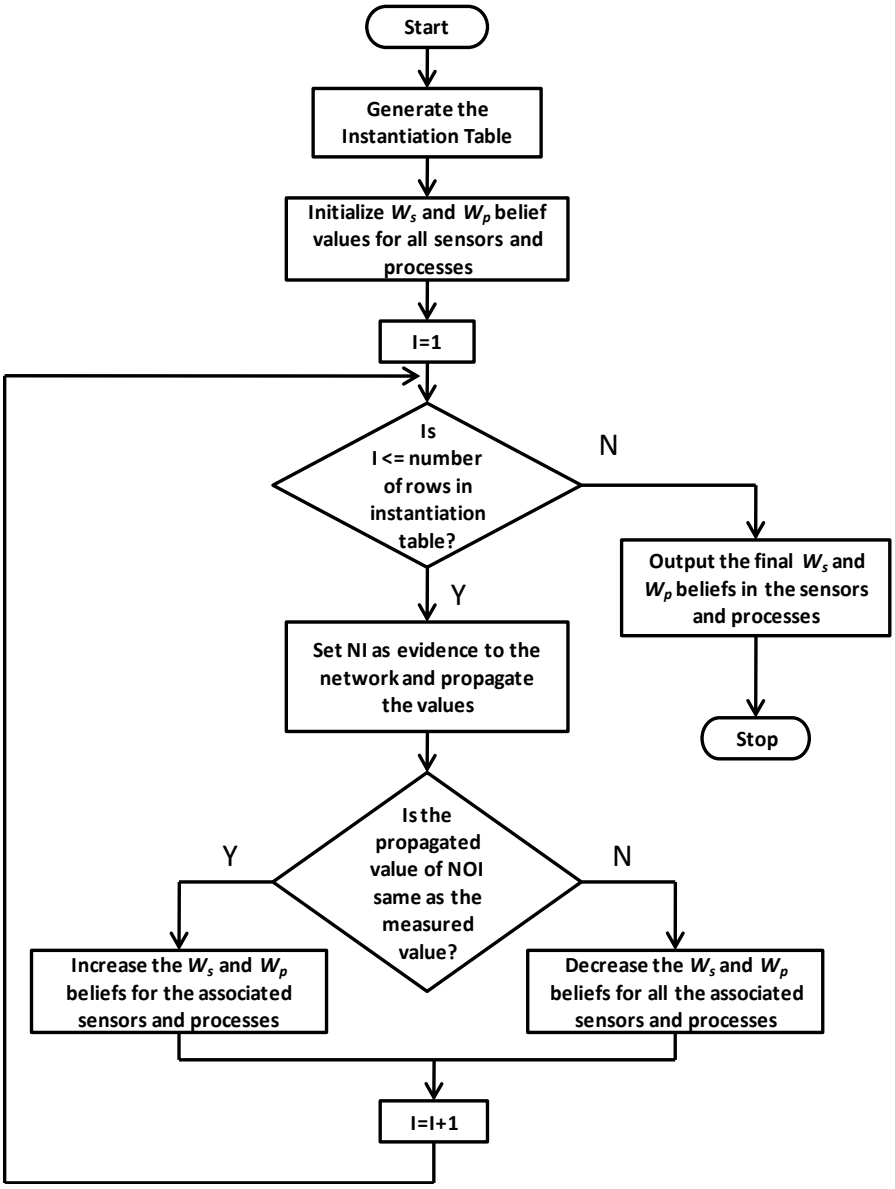


Figure 5-9: Flowchart for the Sensor and Process Fault Detection and Isolation (SPFDI) Algorithm

The final element of the SPFDI algorithm is to decide the amount ε_S and ε_P by which W_S and W_P are modified respectively, after each step in the fault detection and isolation cycle. If the resultant value of the *NoI* (obtained by setting *NI* as evidence) is the same as the actual value indicated by its corresponding sensor, then beliefs for the *NI* and *NoI*

under consideration and all the intermediate *PoI* between the *NI* and *NoI* are increased by their associated ε_S and ε_P values. In the converse situation, all these belief values are decreased by the same amount. The magnitude of ε_S for a particular sensor is determined by the number of times (n_S) it figures either as a *NI* or *NoI* in the instantiation table and is considered a fraction of the initial weight for that sensor. Similarly the magnitude of ε_P for a particular process is determined by the number of times (n_P) it occurs in the instantiation table and is also a fraction of the initial weight for that process[14]. Thus,

$$\varepsilon_S = \frac{W_{S_{initial}}}{n_S} \text{ and } \varepsilon_P = \frac{W_{P_{initial}}}{n_P} \qquad \text{Eq.(5-1)}$$

The final magnitudes of W_S and W_P for each sensor and each process at the end of a cycle will be considered representative of whether a particular sensor or a process is faulty or not. The SPFDI algorithm can now be summarized as follows:

Objective Given a Bayesian network *B* representing set of variables V_i and a set of processes P_j, set of measured sensor readings S_i corresponding to each V_i (i.e. $S_i \equiv V_i$), isolate the cause of undesired deviation in sensor readings

Inputs

- Number of steps in the isolation cycle *N* (number of rows in the instantiation table)
- Number of sensors, *i=1,2,.....l* (number of nodes in the network) and number of processes *j=1,2,.....m* (number of links in the network)
- Nodes instantiated (*NI*, $NI \subseteq V$), node of interest (*NoI*, $NoI \subseteq V$), processes of interest (PoI, $PoI \subseteq P$) at each step in the instantiation table
- n_s for each *V* ($n_s \subseteq N$), n_p for each *P* ($n_p \subseteq N$)

Outcome List of potentially faulty sensors and processes

[14] Eq.(5-1) is valid for $W_{S_{initial}} = W_{P_{initial}} = 0.5$. If other initial belief values are used (for instance, when accurate sensor or process reliability data is available), the equation needs to be modified accordingly so that the condition $0 \leq W_S, W_P \leq 1$ is always satisfied at the end of the fault detection and isolation cycle, with a value of 0 indicating a potential fault and 1 indicating a healthy component.

***Algorithm*:**

Generate the instantiation table

Initialize $W_{Si} = 0.5$ for each V_i and $W_{Pj} = 0.5$ for each P_j

Calculate $\varepsilon_{Si} = 0.5/ n_{Si}$ for each V_i and $\varepsilon_{Pj} = 0.5/n_{Pj}$ for each P_j

Loop through number of steps in the isolation cycle $N_{ic}=1,2,....N$

Loop through $i= 1,2,....l$

If(Node $i \equiv NI$)

Instantiate the state of NI with the corresponding measured value $S_i = s$

Exit loop i

Update beliefs for all nodes in the network

Determine value $V_i = v$ of the NoI based on suitable criterion[15]

Measure the value from the sensor corresponding to NoI as $S_i = s$

If ($V_i = S_i$) for the NoI

Loop through $i= 1,2,....l$

If(Node $i = NI$)

Increase W_{Si} for NI by ε_{Si}

If(Node $i = NoI$)

Increase W_{Si} for NoI by ε_{Si}

Exit loop i

Loop through $j= 1,2,....n$

Increase W_{Pj} for each PoI_j by ε_{Pi}

Exit loop j

else

Loop through $i= 1,2,....l$

If(Node $i = NI$)

Decrease W_{Si} for NI by ε_{Si}

If(Node $i = NoI$)

[15] Most Probable State, Expected Value, etc.

Decrease W_{Si} for NoI by ε_{Si}

Exit loop i

Loop through $j = 1,2,....n$

Decrease W_{Pj} for each PoI_j by ε_{Pi}

Exit loop j

Exit Loop N_{ic}

Loop through number of sensors $i = 1,2,3....l$

Sort W_S descending

Exit loop i

Loop through number of processes $j = 1,2,3.....m$

Sort W_P descending

Exit loop j

Figure 5-10: Sensor and Process Fault Detection and Isolation (SPFDI) Algorithm

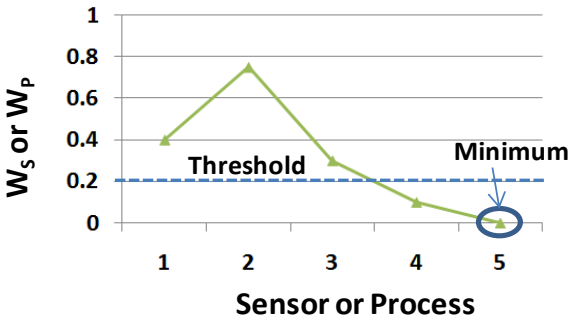


Figure 5-11: Interpreting W_S and W_P values

Figure 5-10 shows the pseudo-code for this algorithm. At the end of a single iteration of this algorithm, the sensor corresponding to the variable with W_{Smin} or the process with W_{Pmin} can be identified as being the potentially faulty. Depending on the application, a suitable threshold may also be defined to indicate the lowest acceptable values for W_S and W_P, below which sensors or processes may be deemed faulty. For instance, when there are multiple sensor faults, comparing the sorted values of W_S against such a threshold should provide an indication of which sensors are most likely to be faulty. This belief is further strengthened if the same results are obtained after multiple iterations of the algorithm.

5.5. EXAMPLE OF SENSOR AND PROCESS FAULT DETECTION AND ISOLATION

Consider the network in Figure 5-8 with the corresponding instantiation table shown in Table 5-2. As before, let the readings indicated by the sensors for the nodes *A*, *B*, *C*, *D* are *A=a*, *B=b*, *C=c*, and *D=d* respectively under regular conditions[16] and a_{inf}, b_{inf}, c_{inf}, d_{inf} be the corresponding values obtained via probabilistic inference using the network. The W_S and W_P values for each of the sensors and processes considered in the instantiation table (Table 5-2) are calculated based on Eq.(5-1) as follows:

$$\varepsilon_A = \frac{0.5}{3} = 0.1667 \qquad \varepsilon_B = \frac{0.5}{3} = 0.1667 \qquad \varepsilon_C = \frac{0.5}{3} = 0.1667 \qquad \varepsilon_D = \frac{0.5}{3} = 0.1667$$

$$\varepsilon_{A->B} = \frac{0.5}{3} = 0.1667 \qquad \varepsilon_{B->C} = \frac{0.5}{4} = 0.125 \qquad \varepsilon_{C->D} = \frac{0.5}{3} = 0.1667$$

5.5.1. Case 1: Sensor Fault

Suppose the readings indicated by the sensors are *A=a*, *B=b'≠b* (where *b'* indicates that, due to a sensor fault, the state of *B* corresponding to the value *b'* is different from the state of *B* corresponding to the value *b*), *C=c*, and *D=d* respectively.

The algorithm starts with an assumption of ignorance regarding the status of any sensor or process. This implies that there is an equal chance that any of the sensors or processes could be at fault. Therefore the initial beliefs in the different sensors and processes are:

$$W_A = 0.5 \quad W_B = 0.5 \quad W_C = 0.5 \quad W_D = 0.5 \quad W_{A->B} = 0.5 \quad W_{B->C} = 0.5 \quad W_{C->D} = 0.5$$

Since the CPT parameters in the network are still unchanged after the last update when the all the sensors and processes were deemed to be working correctly, after the first instantiation, the value b_{inf} will not be the same as *b'*. Hence, the beliefs in the sensors for *A* and *B* and the process *A->B* will be decreased by their corresponding ε values. The revised belief values now become:

$$W_A = 0.5\text{-}0.1667 = 0.3333 \qquad W_B = 0.5\text{-}0.1667 = 0.3333 \qquad W_C = 0.5 \qquad W_D = 0.5$$

$$W_{A->B} = 0.5\text{-}0.1667 = 0.3333 \qquad W_{B->C} = 0.5 \qquad W_{C->D} = 0.5$$

[16] *A=a, B=b…..* implies that the value *a* correspond to one of the states of *A,* and so on.

After the second instantiation, since the sensor for C is not faulty and the processes A->B and B->C are also not faulty (as determined by the previous fault detection and isolation cycle), the value obtained after probabilistic propagation i.e. c_{inf} is expected to be the same as the reading c from the sensor for node C. Thus, the beliefs in the sensors A and C and the beliefs in the intermediate the processes A->B and B->C are increased by their respective ε values. The revised beliefs after the second instantiation in the cycle are:

$W_A = 0.3333+0.1667 = 0.5$ $W_B = 0.3333$ $W_C = 0.5+0.1667 = 0.6667$ $W_D = 0.5$

$W_{A->B} = 0.3333+0.1667 = 0.5$ $W_{B->C} = 0.5 +0.125 = 0.625$ $W_{C->D} = 0.5$

Repeating the above procedure for each step in the instantiation table, the results obtained are shown in Table 5-3. Thus, after the complete cycle, the belief in the sensor for node B is zero whereas the belief in the other sensors and processes are higher, as they should be.

Table 5-3: Change in W_S and W_P Values with a Faulty Sensor

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.3333	0.3333	0.5	0.5	0.3333	0.5	0.5
2	0.5	0.3333	0.6667	0.5	0.5	0.625	0.5
3	0.6667	0.3333	0.6667	0.6667	0.6667	0.750	0.6667
4	0.6667	0.1667	0.5	0.6667	0.6667	0.625	0.6667
5	0.6667	0.0000	0.5	0.5	0.6667	0.5	0.5
6	0.6667	**0.0000**	0.6667	0.6667	0.6667	0.5	0.6667

These beliefs are calculated after a single sample of data. If the same result is obtained in consecutive isolation cycles, then it is indicative of a confirmed fault in the sensor for node B[17]. However, at this point, these values may be used by the system operator to decide whether or not to continue using the data from the sensor for node B.

[17] The number of cycles required is based on the application requirement and operator judgment. Once a sensor fault has been determined with certainty, the operator can also choose to modify the instantiation table by eliminating the steps involving the sensor for node B either as an instantiated node or as a node of interest (steps 1, 4, and 5). Although this does not provide any additional information, with the remaining instantiations, since all the other sensors and processes are operating correctly, the corresponding belief values will ideally attain a value of 1 in the subsequent isolation cycles.

5.5.2. Case 2: Process Fault

Since the Bayesian network represents the causal relations between all the variables of interest in the system, in case of a process fault, the effect of the fault is noticed in all the variables downstream of the process and not confined just to the variables in the process itself. This is manifested as a variation in the readings of all the associated sensors. In other words, a fault in the process *B->C* will be reflected as deviations in the readings of the sensors for nodes *B*, *C* and *D* from their ideal values (as obtained from the network). Thus, in this case, after the first instantiation in the instantiation table, since the process *A->B* is not faulty and the sensors for *A* and *B* are also not faulty, the value b_{inf} will agree with the reading *b* from the sensor for node *B*. Thus the beliefs in the sensors and processes are revised as follows:

$$W_A = 0.5+0.1667 = 0.6667 \qquad W_B = 0.5+0.1667 = 0.6667 \qquad W_C = 0.5 \qquad W_D = 0.5$$

$$W_{A->B} = 0.5+0.1667 = 0.6667 \qquad W_{B->C} = 0.5 \qquad W_{C->D} = 0.5$$

In the next four instantiations, when *A* or *B* is the instantiated node, since the faulty process *B->C* is involved as a process of interest, the values *c* and *d* indicated by the sensors will be different from the inferred values c_{inf} and d_{inf}. In the last step of the isolation cycle, only sensors for *C* and *D* and the process *C->D* are involved. Since none of these components are faulty, the measured and the inferred values *d* and d_{inf} are found to be in agreement. The variation in all the beliefs in this scenario is shown in Table 5-4.

Table 5-4: Change in W_S and W_P Values with a Faulty Process

	Sensors				**Processes**		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.6667	0.6667	0.5	0.5	0.6667	0.5	0.5
2	0.5	0.6667	0.3333	0.5	0.5	0.375	0.5
3	0.3333	0.6667	0.3333	0.3333	0.3333	0.25	0.3333
4	0.3333	0.5	0.1667	0.3333	0.3333	0.125	0.3333
5	0.3333	0.3333	0.1667	0.1667	0.3333	0.0000	0.1667
6	0.3333	0.3333	0.3333	0.3333	0.3333	**0.0000**	0.3333

It is observed that the belief in the process *B->C* is reduced to zero whereas the beliefs in all the other sensors and processes are higher. These belief values can be used

to alert the operator that a potential fault may exist in the system. As in the earlier case, the process *B->C* can be declared as being faulty with certainty only if the same results are obtained after multiple samples have been analyzed (i.e. after a certain number of fault detection and isolation cycles). Thus, the algorithm is able to correctly distinguish between a sensor and a process fault (i.e. it does not interpret the deviations in the sensors corresponding to the nodes *B*, *C* and *D* as multiple sensor faults). The final belief values are indicative of the trustworthiness of a specific sensor or process. This knowledge can be immensely useful when updating the model parameters (i.e. CPT values of the various nodes) and eventually the stored performance maps.

5.5.3. Special Cases

The methodology described above can also be used to isolate multiple sensor faults as well. Consider an example where the sensors for nodes *B* and *C* are both faulty but the sensors for *A* and *D* are operating normally and there are no faults in any of the intermediate processes. In this case, the variation in the beliefs following the instantiation table (Table 5-2) is shown in Table 5-5.

Table 5-5: Change in W_S and W_P Values with Multiple Sensor Faults

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.3333	0.3333	0.5	0.5	0.3333	0.5	0.5
2	0.1667	0.3333	0.3333	0.5	0.1667	0.375	0.5
3	0.3333	0.3333	0.3333	0.6667	0.3333	0.5	0.6667
4	0.3333	0.1667	0.1667	0.6667	0.3333	0.375	0.6667
5	0.3333	0.0000	0.1667	0.5	0.3333	0.25	0.5
6	0.3333	**0.0000**	**0.0000**	0.3333	0.3333	0.25	0.3333

At the end of the fault detection and isolation cycle, it can be observed that the belief values for the sensors corresponding to nodes *B* and *C* are zero whereas the beliefs in the other sensors and processes are higher. There are however certain scenarios in which the algorithm produces inconsistent or incorrect results. With any unforeseen deviation in the values from a sensor corresponding to a root node in the network,

applying the algorithm results in the associated sensor as well as all the processes that originate from that node (relating the node to its immediate child nodes) identified as being faulty[18].

Table 5-6: Change in W_S and W_P Values with a Fault in a Sensor Corresponding to a Root Node

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.3333	0.3333	0.5	0.5	0.3333	0.5	0.5
2	0.1667	0.3333	0.333	0.5	0.1667	0.375	0.5
3	0.0000	0.3333	0.3333	0.3333	0.0000	0.25	0.3333
4	0.0000	0.5	0.5	0.3333	0.0000	0.375	0.3333
5	0.0000	0.6667	0.5	0.5	0.0000	0.5	0.5
6	**0.0000**	0.6667	0.6667	0.6667	**0.0000**	0.625	0.6667

Table 5-7: Change in W_S and W_P Values with a Fault in a Sensor Corresponding to a Leaf Node

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.6667	0.6667	0.5	0.5	0.6667	0.5	0.5
2	0.8334	0.6667	0.6667	0.5	0.8334	0.6667	0.5
3	0.6667	0.6667	0.6667	0.3333	0.6667	0.5	0.3333
4	0.6667	0.8334	0.8334	0.3333	0.6667	0.6667	0.3333
5	0.6667	0.6667	0.8334	0.1667	0.6667	0.8334	0.1667
6	0.6667	0.6667	0.6667	**0.0000**	0.6667	0.6667	**0.0000**

In the case of a deviation in the readings from a sensor corresponding to a leaf node, the algorithm is again not completely reliable in identifying whether the departure from the expected values is due to a fault in the sensor corresponding to the node or if it is the result of a change in one or more of the processes linking the leaf node to its immediate parents. These two cases are shown in Table 5-6 and Table 5-7 respectively which show the variation in beliefs when there is a fault in the sensor corresponding to node *A* and node *D* respectively. There are also some additional cases wherein the

[18] This may be partially attributed to instantiating only ancestral nodes. With the present algorithm and network structure, the root nodes are never examined as a *NoI*.

algorithm may produce inconsistent results. The following section aims to provide an explanation for all these inconsistencies.

5.5.3.1.The Subset Problem

Before delving into details, the terms '*subset*' and '*superset*' as used in the context of this discussion need to be defined using a simple example. In Figure 5-12, for every shaded cell in column *X*, the corresponding cell in the same row of column *Y* is also shaded, but column *Y* has other additional shaded cells. Column *X* will be referred to as a subset of column *Y* ($X \subseteq Y$) or *Y* can be said to be a superset of *X* ($Y \supseteq X$). In a similar vein, the column *Y* is a subset of the union of columns *X* and *Z* ($Y \subseteq X \cup Z$) i.e. for every shaded cell in column *Y,* a corresponding cell in the same row is also shaded in either column *X* or column *Z*[19]. These concepts can now be extended to an instantiation table to explain the results in the previous section.

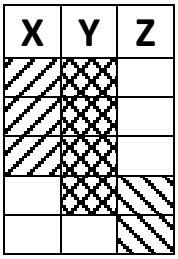


Figure 5-12: Concept of Subset

Consider the instantiation table in Table 5-8. Let Col_S and Col_P represent the columns in the instantiation table corresponding to a particular sensor or process, where $S = \{A, B, C, D\}$ and $P = \{A->B, B->C, C->D\}$. In each step of the fault detection and isolation cycle where the root node *A* is involved as *NI*, the process relating *A* to its immediate child *B* i.e. the link *A->B* is also involved in that step, irrespective of the *NoI* (with *A* as *NI* the nodes *B*, *C*, *D* are *NoI* in rows 1-3 of the table).

This is observed in Table 5-8 where the shaded cells in $Col_{A->B}$ and Col_A are the same. It can be said that $Col_{A->B} \subseteq Col_A$. Thus, a subset may be formed between the columns corresponding to a root node and a process originating from that node, when the root node is considered as *NI* and its descendents are sequentially considered as *NoI,* with the process always included in the path between the *NI* and the *NoI* at every instantiation.

[19] The term 'subset' is used to also include the case when the number of shaded cells in any two columns are identical

Such overlap results in W_A and $W_{A->B}$ to be incremented or decremented by the same amount throughout the cycle (n_S and n_P are the same, resulting in the same values of ε_S and ε_P).

Table 5-8: Instantiation Table for the Network *A -> B -> C -> D*

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
1							
2							
3							
4							
5							
6							

Nodes Instantiated **Node of Interest** **Process of Interest**

Similarly, it is observed that $Col_{C->D} \subseteq Col_D$ i.e. at every step when node *D* is the *NoI* then the process *C->D* that terminates at that node is also involved. Thus, a subset may be formed between the columns corresponding to a terminal leaf node and a process terminating at that node, when the leaf node is considered as *NoI* and its different ancestral nodes are considered as *NI,* with the process again always included in the path between the *NI* and the *NoI* at every instantiation.

There may be other cases when columns corresponding to certain sensors or processes in the network (say, Col_{S-I} and Col_{P-I}) form a subset of the columns that correspond to other sensors or processes that occur in the network (say, Col_{S-II} and Col_{P-II}), either individually or by considering a union of those columns. For instance, some possible combinations could be $Col_{S-I} \subseteq Col_{S-II}$ or $Col_{S-I} \subseteq Col_{P-II}$ or $Col_{P-I} \subseteq Col_{S-II}$ or $Col_{P-I} \subseteq Col_{P-II}$ or $Col_{S-I} \subseteq Col_{S-II} \cup Col_{P-II}$ or or $Col_{P-I} \subseteq Col_{S-II} \cup Col_{P-II}$ or $Col_{S-I} \subseteq Col_{P-I} \cup Col_{P-II}$, etc. Note that, in each combination, the union is typically defined for a column corresponding to a sensor and a process (or set of processes) that terminates or originates at the node corresponding to that sensor in the network.

For the instantiation table (Table 5-8), consider a case where the process *B->C* is faulty and there is also a fault in the sensor corresponding to node *C*. It is seen that

$Col_D \subseteq (Col_C \cup Col_{B->C})$ and $Col_{C->D} \subseteq (Col_C \cup Col_{B->C})$. In this case, the change in the beliefs for all the sensors and processes are shown in Table 5-9. It is observed that in addition to faults being signaled for the process *B->C* and the sensor corresponding to node *C*, the final belief values result in false alarms with the sensor corresponding to node *D* and the *C->D* process also being diagnosed as faulty.

Table 5-9: Change in W_S and W_P Values when Columns in the Instantiation Table Form Subsets

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
1	0.6667	0.6667	0.5	0.5	0.6667	0.5	0.5
2	0.5	0.6667	0.3333	0.5	0.5	0.375	0.5
3	0.3333	0.6667	0.3333	0.3333	0.3333	0.25	0.3333
4	0.3333	0.5	0.1667	0.3333	0.3333	0.125	0.3333
5	0.3333	0.3333	0.1667	0.1667	0.3333	0.0000	0.1667
6	0.3333	0.3333	**0.0000**	**0.0000**	0.3333	**0.0000**	**0.0000**

In each of the cases described above, if a fault occurs in a sensor or a process whose columns form the superset of columns corresponding to other sensors or processes in the network, then the algorithm also incorrectly flags the sensors or processes corresponding to columns in the subset as faulty. This situation is termed as the '*subset problem'*. As evident from the discussion above, the structure of the network plays an important role in this issue. Some network structures may reduce the possibility of such subsets being formed whereas others may worsen the problem[20]. The extent to which the subset problem occurs determines the ability of the SPFDI algorithm to uniquely identify and isolate different sources of faults. This aspect is discussed further in Section 5.7.1.

5.5.4. Use of Redundant Nodes

To enable the SPFDI algorithm to overcome the complications created by the subset problem, it is desirable to eliminate the possibilities of subsets being formed as far

[20] The practice of instantiating only the ancestral nodes (in the present version of the algorithm) may also a contributing factor to the subset problem. Future research in exploring additional types of instantiations may help alleviate this issue to some extent.

as possible (especially if a particular sensor or a process is critical from the application point of view). One way to do this would be to explore additional instantiations using abductive inferencing, which would add additional possibilities for the *NI* and *NoI* thereby resulting in additional rows in the instantiation table which would make every column of Table 5-8 distinct. Another possible approach to resolve these issues is by using redundant sources of information for the root and leaf nodes in order to distinguish between sensor and process faults.

In the simplest form, these redundant nodes can represent additional physical sensors for the same measurand. For instance two sensors may be used to measure the variable *D*, one of which is highly accurate and is relied upon as the primary sensor used to update the performance maps and for decision-making (say, the sensor corresponding to node *D* in Figure 5-13) and the second sensor may be somewhat less accurate but can still be used to corroborate the health of the first sensor (for instance, the sensor corresponding to node D_2 in Figure 5-13). The redundant nodes could also represent other sources of information like additional performance maps for the same measurand, or an expert operator's judgment of its value based on experience, etc.

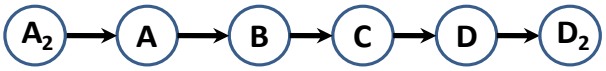


Figure 5-13: Addition of Redundant Nodes

	$S=Value_1$	$S=Value_2$	$S=Value_3$ .....
$S_2=Value_1$	1	0	0 .....
$S_2=Value_2$	0	1	0 .....
$S_2=Value_3$	0	0	1 .....
⋮	⋮	⋮	

Figure 5-14 : CPTs for Redundant Nodes

Figure 5-13 shows the addition of two such redundant nodes, representing additional sensors corresponding to the variables *A* and *D*, respectively. Since the sensors corresponding to the redundant nodes measure the same variables as the sensors corresponding to the nodes in the original network, the CPT for these nodes (D_2 and *A* in the present case) takes a form similar to an identity matrix. In Figure 5-13, the sensors corresponding to nodes *A* and A_2 measure the same physical variable. Hence, the

process A_2->A does not represent a real physical process and as such can be considered to be a 'perfect' process which never fails. With the additional nodes, more instantiations can be introduced to make the columns in the instantiation table distinct. This would allow nodes which were formerly root nodes to be now considered as *NoI* during the fault detection and isolation cycle. The same holds true for the former leaf nodes which now have additional child nodes that can now be considered as *NoI*. With the additional sensors and intermediate processes, the list of additional instantiations[21] that can now be added are shown in Table 5-10.

Table 5-10: Instantiation Table with the Addition of Redundant Nodes

	Sensors						**Processes**				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
1	*NI*	*NoI*					*PoI*				
2	*NI*		*NoI*				*PoI*	*PoI*			
3	*NI*			*NoI*			*PoI*	*PoI*	*PoI*		
4	*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*	
5	*NI*					*NoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*
6		*NI*	*NoI*					*PoI*			
7		*NI*		*NoI*				*PoI*	*PoI*		
8		*NI*			*NoI*			*PoI*	*PoI*	*PoI*	
9		*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*
10			*NI*	*NoI*					*PoI*		
11			*NI*		*NoI*				*PoI*	*PoI*	
12			*NI*			*NoI*			*PoI*	*PoI*	*PoI*
13				*NI*	*NoI*					*PoI*	
14				*NI*		*NoI*				*PoI*	*PoI*
15					*NI*	*NoI*					*PoI*

The W_S and W_P values for the sensors and processes are calculated as before. However, in modifying the values W_P for the process relating the redundant nodes, there is a crucial difference. Since such processes are assumed to be perfect, their belief values are always set to 1. Note that the columns for the sensor corresponding to the node *A* and the process *A->B* are now distinct. Thus, with any unforeseen deviations in the readings from the sensor corresponding to node *A*, it is now possible for the algorithm to isolate the fault in

[21] With *NS* number of sensors in the network a total of (*NS*-1)! instantiations are possible (the number of rows in the instantiation table)

the sensor *A* without misdiagnosing a fault in the *A->B* process. The change in the belief values are shown in Table 5-11. Similarly with any unforeseen deviation in the readings from the sensor corresponding to node *D*, the algorithm can isolate the cause of deviation to a faulty sensor or to a fault in the *C->D* process.

Table 5-11: Change in W_S and W_P Values with a Faulty Sensor and the Addition of Redundant Nodes

	Sensors						**Processes**				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
0	0.5	0.5	0.5	0.5	0.5	0.5	1.000	0.5	0.5	0.5	1.000
1	0.4	0.4	0.5	0.5	0.5	0.5	1.000	0.5	0.5	0.5	1.000
2	0.5	0.4	0.6	0.5	0.5	0.5	1.000	0.5625	0.5	0.5	1.000
3	0.6	0.4	0.6	0.6	0.5	0.5	1.000	0.625	0.5556	0.5	1.000
4	0.7	0.4	0.6	0.6	0.6	0.5	1.000	0.6875	0.6112	0.5625	1.000
5	0.8	0.4	0.6	0.6	0.6	0.6	1.000	0.75	0.6666	0.625	1.000
6	0.8	0.3	0.5	0.6	0.6	0.6	1.000	0.6875	0.6666	0.625	1.000
7	0.8	0.2	0.5	0.5	0.6	0.6	1.000	0.625	0.6112	0.625	1.000
8	0.8	0.1	0.5	0.5	0.5	0.6	1.000	0.5625	0.5556	0.5625	1.000
9	0.8	0.0	0.5	0.5	0.5	0.5	1.000	0.5	0.5	0.5	1.000
10	0.8	0.0	0.6	0.6	0.5	0.5	1.000	0.5	0.5556	0.5	1.000
11	0.8	0.0	0.7	0.6	0.6	0.5	1.000	0.5	0.6112	0.5625	1.000
12	0.8	0.0	0.8	0.6	0.6	0.6	1.000	0.5	0.6666	0.625	1.000
13	0.8	0.0	0.8	0.7	0.7	0.6	1.000	0.5	0.6666	0.6875	1.000
14	0.8	0.0	0.8	0.8	0.7	0.7	1.000	0.5	0.6666	0.75	1.000
15	0.8	**0.0**	0.8	0.8	0.8	0.8	1.000	0.5	0.6666	0.75	1.000

Note that even though the columns corresponding to the sensor for node A_2 and the process $A->A_2$ are still identical, because of the assumption of an ideal process, a fault in the sensor corresponding to A_2 can still be isolated ($W_{A->A2}$ remains unchanged throughout the fault detection and isolation cycle). With the additional nodes in the network $Col_D \nsubseteq (Col_C \cup Col_{B->C})$ but $Col_{C->D} \subseteq (Col_C \cup Col_{B->C})$. Hence, again considering the example where process *B->C* and the sensor corresponding to node *C* are both faulty, the algorithm would no longer flag the sensor corresponding to node *D* as being faulty but would still wrongly identify the *C->D* process as being faulty.

Also from Table 5-10, it is seen that $Col_B \subseteq (Col_{A->B} \cup Col_{B->C})$. Thus, if multiple process faults exist in the processes *A->B* and *B->C,* the SPFDI algorithm would misdiagnose the sensor corresponding to node *B* as faulty. In all such cases, the only

recourse with the present version of the algorithm is to introduce additional instantiations to make the columns in the instantiation table completely distinct. This may be done as demonstrated above by including redundant nodes for the appropriate nodes in the network (say, node B_2 representing a sensor in addition to the original sensor corresponding to node B, in the latter example).

5.6. APPLICATION OF THE SENSOR AND PROCESS FAULT DETECTION AND ISOLATION ALGORITHM FOR THE TEST SETUP

Figure 5-15 represents a Bayesian network of the test set up used in this work (a detailed description and relevant specifications of the various components is presented in Appendix B). It represents a simplified version of the detailed network developed in Chapter 3, using some of the criteria presented in Chapter 4 (such as the sensors available, memory and computational requirements, etc.).

The nodes *DC* and *FR* represent the duty cycle and frequency of the PWM signal used to modulate the supply voltage V_s. These are the control parameters set by the operator and sent as commands to the H-bridge amplifiers to control the motor terminal voltage. The nodes *V* and *I* represent the total voltage applied and the total current drawn across all the four motor phases at any instant. The node τ represents the total torque produced by the motor. The node *S* is the motor speed (under known loads acting on the motor shaft and applied using a hysteresis brake). The node *N* represents the acoustic noise generated by the motor.

Suppose the control parameters are set at values of *DC* = 1% and *FR*= 5 kHz for a constant supply voltage of V_s =20V, then the voltage available across the motor terminals should ideally be 0.2 V, which should be the reading indicated by the voltage sensor. The corresponding values of current (1.2 A), torque (0.8 Nm), speed (400 rpm) and noise (74 dB) under these conditions are shown in Figure 5-15.

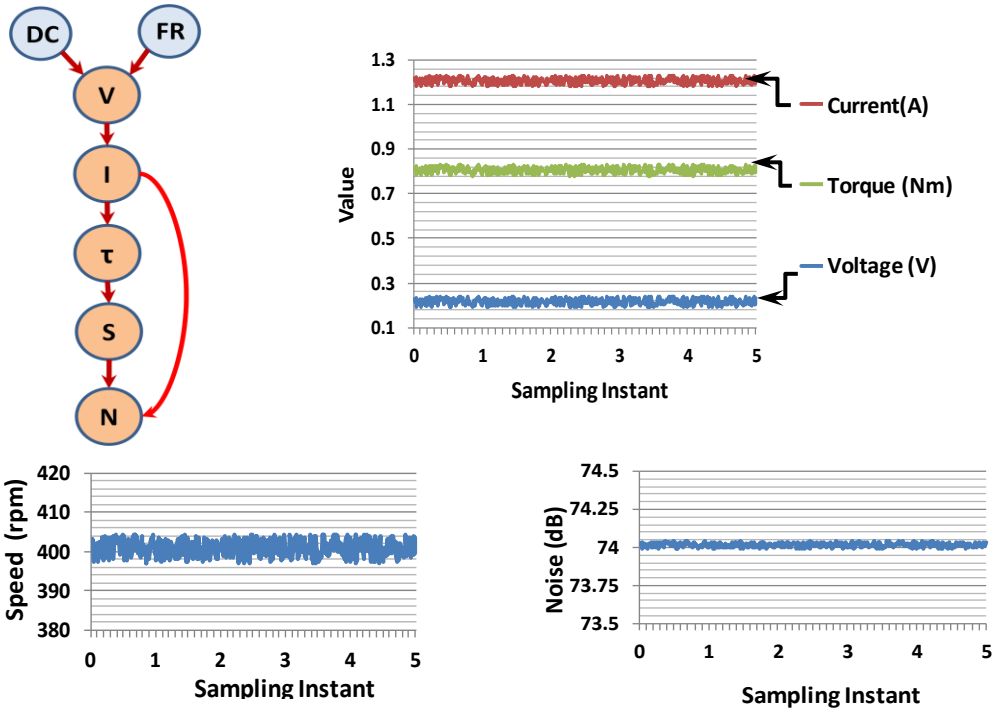


Figure 5-15: Simplified Bayesian Network of the Testbed and Sensor Values

Table 5-12: Instantiation Table for the Testbed Bayesian Network

	Sensors							Processes						
Step	DC	FR	V	I	T	S	N	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N
1														
2														
3														
4														
5														
6														
7														
8														
9														
10														
11														
12														
13														
14														
15														

Nodes Instantiated Node of Interest Process of Interest

The various steps in the fault detection and isolation cycle for the above network are as shown in Table 5-12. Note that Col_{DC}, Col_{FR}, $Col_{DC->V}$ and $Col_{FR->V}$ are all similar. Col_N, $Col_{I->N}$ and $Col_{S->N}$ also form subsets. The variation in the weights W_S and W_P for the different sensors and processes are shown in Table 5-13. When all the sensors are operating correctly and there is no change in the relationships between the different variables, all the belief values tend to a value of 1 (row 15).

Table 5-13: Variation in W_S and W_P Values when all Sensors and Processes are Operating Correctly

	Sensors							**Processes**						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.600	0.600	0.600	0.500	0.500	0.500	0.500	0.600	0.600	0.500	0.500	0.500	0.500	0.500
2	0.700	0.700	0.600	0.571	0.500	0.500	0.500	0.700	0.700	0.563	0.500	0.500	0.500	0.500
3	0.800	0.800	0.600	0.571	0.600	0.500	0.500	0.800	0.800	0.625	0.556	0.500	0.500	0.500
4	0.900	0.900	0.600	0.571	0.600	0.600	0.500	0.900	0.900	0.688	0.611	0.500	0.563	0.500
5	1.000	1.000	0.600	0.571	0.600	0.600	0.600	1.000	1.000	0.750	0.667	0.600	0.625	0.600
6	1.000	1.000	0.700	0.643	0.600	0.600	0.600	1.000	1.000	0.813	0.667	0.600	0.625	0.600
7	1.000	1.000	0.800	0.643	0.700	0.600	0.600	1.000	1.000	0.875	0.722	0.600	0.625	0.600
8	1.000	1.000	0.900	0.643	0.700	0.700	0.600	1.000	1.000	0.938	0.778	0.600	0.688	0.600
9	1.000	1.000	1.000	0.643	0.700	0.700	0.700	1.000	1.000	1.000	0.833	0.700	0.750	0.700
10	1.000	1.000	1.000	0.714	0.800	0.700	0.700	1.000	1.000	1.000	0.889	0.700	0.750	0.700
11	1.000	1.000	1.000	0.786	0.800	0.800	0.700	1.000	1.000	1.000	0.944	0.700	0.813	0.700
12	1.000	1.000	1.000	0.857	0.800	0.800	0.800	1.000	1.000	1.000	1.000	0.800	0.875	0.800
13	1.000	1.000	1.000	0.857	0.900	0.900	0.800	1.000	1.000	1.000	1.000	0.800	0.938	0.800
14	1.000	1.000	1.000	0.929	1.000	0.900	0.900	1.000	1.000	1.000	1.000	0.900	1.000	0.900
15	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

Now, consider a scenario where there is some factor affecting the behavior of the voltage sensor (e.g. a degradation in the sensor core causing a bias fault in the sensor) and it indicates a value other than 0.2 V (say, 0.4 V as shown in Figure 5-16). All the other sensors continue to indicate their expected values (Figure 5-15. In this case, only the W_s corresponding to the voltage sensor decreases to zero (Table 5-14), indicating that the likely source of deviation in the voltage reading is due to a potential issue with its sensor. It can also be seen that the weights for all the other sensors as well as the processes remains unchanged or have increased from their initial value. If the deviation in the

voltage reading was due to an increased supply voltage level (which would be considered as a 'fault' under the present terminology), then it would be reflected in the form of potentially increased readings in all the other variables as well.

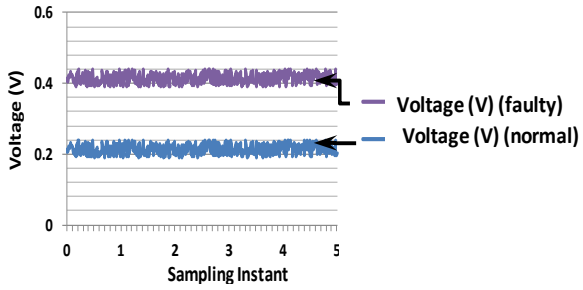


Figure 5-16: Faulty Voltage Sensor

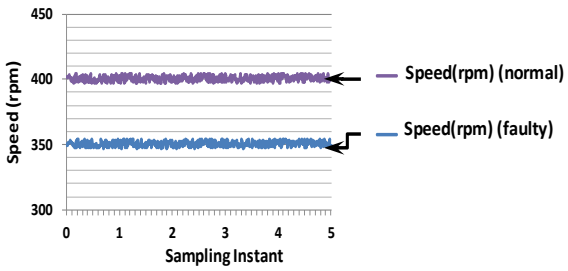


Figure 5-17: Faulty Speed Sensor

Table 5-14: Variation in W_s and W_p values with a faulty voltage sensor

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.400	0.400	0.400	0.500	0.500	0.500	0.500	0.400	0.400	0.500	0.500	0.500	0.500	0.500
2	0.500	0.500	0.400	0.571	0.500	0.500	0.500	0.500	0.500	0.563	0.500	0.500	0.500	0.500
3	0.600	0.600	0.400	0.571	0.600	0.500	0.500	0.600	0.600	0.625	0.556	0.500	0.500	0.500
4	0.700	0.700	0.400	0.571	0.600	0.600	0.500	0.700	0.700	0.688	0.611	0.500	0.563	0.500
5	0.800	0.800	0.400	0.571	0.600	0.600	0.600	0.800	0.800	0.750	0.667	0.600	0.625	0.600
6	0.800	0.800	0.300	0.500	0.600	0.600	0.600	0.800	0.800	0.688	0.667	0.600	0.625	0.600
7	0.800	0.800	0.200	0.500	0.500	0.600	0.600	0.800	0.800	0.625	0.611	0.600	0.625	0.600
8	0.800	0.800	0.100	0.500	0.500	0.500	0.600	0.800	0.800	0.563	0.556	0.600	0.563	0.600
9	0.800	0.800	0.000	0.500	0.500	0.500	0.500	0.800	0.800	0.500	0.500	0.500	0.500	0.500
10	0.800	0.800	0.000	0.571	0.600	0.500	0.500	0.800	0.800	0.500	0.556	0.500	0.500	0.500
11	0.800	0.800	0.000	0.643	0.600	0.600	0.500	0.800	0.800	0.500	0.611	0.500	0.563	0.500
12	0.800	0.800	0.000	0.714	0.600	0.600	0.600	0.800	0.800	0.500	0.667	0.600	0.625	0.600
13	0.800	0.800	0.000	0.714	0.700	0.700	0.600	0.800	0.800	0.500	0.667	0.600	0.688	0.600
14	0.800	0.800	0.000	0.786	0.800	0.700	0.700	0.800	0.800	0.500	0.667	0.700	0.750	0.700
15	0.800	0.800	**0.000**	0.857	0.800	0.800	0.800	0.800	0.800	0.500	0.667	0.800	0.750	0.800

Consider another example with a fault in the motor speed (calculated from the encoder readings) (Figure 5-17); instead of an expected value of 400 rpm, the indicated speed is

350 rpm and the remaining sensors are operating correctly. In this case, only the W_s value for the speed sensor decreases to zero (Table 5-15).

Table 5-15: Variation in W_S and W_P Values with a Faulty Speed Sensor

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.600	0.600	0.600	0.500	0.500	0.500	0.500	0.600	0.600	0.500	0.500	0.500	0.500	0.500
2	0.700	0.700	0.600	0.571	0.500	0.500	0.500	0.700	0.700	0.563	0.500	0.500	0.500	0.500
3	0.800	0.800	0.600	0.571	0.600	0.500	0.500	0.800	0.800	0.625	0.556	0.500	0.500	0.500
4	0.700	0.700	0.600	0.571	0.600	0.400	0.500	0.700	0.700	0.563	0.500	0.500	0.438	0.500
5	0.800	0.800	0.600	0.571	0.600	0.400	0.600	0.800	0.800	0.625	0.556	0.600	0.500	0.600
6	0.800	0.800	0.700	0.643	0.600	0.400	0.600	0.800	0.800	0.688	0.556	0.600	0.500	0.600
7	0.800	0.800	0.800	0.643	0.700	0.400	0.600	0.800	0.800	0.750	0.611	0.600	0.500	0.600
8	0.800	0.800	0.700	0.643	0.700	0.300	0.600	0.800	0.800	0.688	0.556	0.600	0.438	0.600
9	0.800	0.800	0.800	0.643	0.700	0.300	0.700	0.800	0.800	0.750	0.611	0.700	0.500	0.700
10	0.800	0.800	0.800	0.714	0.800	0.300	0.700	0.800	0.800	0.750	0.667	0.700	0.500	0.700
11	0.800	0.800	0.800	0.643	0.800	0.200	0.700	0.800	0.800	0.750	0.611	0.700	0.438	0.700
12	0.800	0.800	0.800	0.714	0.800	0.200	0.800	0.800	0.800	0.750	0.667	0.800	0.500	0.800
13	0.800	0.800	0.800	0.714	0.700	0.100	0.800	0.800	0.800	0.750	0.667	0.800	0.438	0.800
14	0.800	0.800	0.800	0.786	0.800	0.100	0.900	0.800	0.800	0.750	0.667	0.900	0.500	0.900
15	0.800	0.800	0.800	0.714	0.800	**0.000**	0.800	0.800	0.800	0.750	0.667	0.800	0.500	0.800

Similar results are observed even with multiple faulty sensors. Table 5-16 shows an example with faulty voltage and speed sensors. Table 5-17 shows an example with an additional faulty torque sensor[22]. In both these cases, it can be seen that deviation in the sensor readings is correctly identified as being caused by a potentially faulty sensor instead of being classified as a process fault. Note that, with multiple faulty sensors, the W_S values for all the faulty sensors does not reduce to zero. However, these values are still relatively low as compared to the W_S or W_P values for the other operational sensors and processes, indicating the possibility of a potential fault in these components. As the number of faulty sensors increases, it is observed that the potential for false alarms also increases. In simulations with various such combinations of faulty sensors, it was found

[22] The torque sensor is assumed to indicate a value of 1Nm as compared to the expected value of 0.8 Nm as shown in Figure 5-15

Table 5-16: Variation in W_S and W_P Values with Faulty Voltage and Speed Sensors

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.400	0.400	0.400	0.500	0.500	0.500	0.500	0.400	0.400	0.500	0.500	0.500	0.500	0.500
2	0.500	0.500	0.400	0.583	0.500	0.500	0.500	0.500	0.500	0.563	0.500	0.500	0.500	0.500
3	0.600	0.600	0.400	0.583	0.600	0.500	0.500	0.600	0.600	0.625	0.556	0.500	0.500	0.500
4	0.500	0.500	0.400	0.583	0.600	0.400	0.500	0.500	0.500	0.563	0.500	0.500	0.438	0.500
5	0.600	0.600	0.400	0.583	0.600	0.400	0.600	0.600	0.600	0.625	0.556	0.600	0.500	0.600
6	0.600	0.600	0.300	0.500	0.600	0.400	0.600	0.600	0.600	0.563	0.556	0.600	0.500	0.600
7	0.600	0.600	0.200	0.500	0.500	0.400	0.600	0.600	0.600	0.500	0.500	0.600	0.500	0.600
8	0.600	0.600	0.100	0.500	0.500	0.300	0.600	0.600	0.600	0.438	0.444	0.600	0.438	0.600
9	0.600	0.600	0.000	0.500	0.500	0.300	0.500	0.600	0.600	0.375	0.389	0.500	0.375	0.500
10	0.600	0.600	0.000	0.583	0.600	0.300	0.500	0.600	0.600	0.375	0.444	0.500	0.375	0.500
11	0.600	0.600	0.000	0.500	0.600	0.200	0.500	0.600	0.600	0.375	0.389	0.500	0.313	0.500
12	0.600	0.600	0.000	0.583	0.600	0.200	0.600	0.600	0.600	0.375	0.444	0.600	0.375	0.600
13	0.600	0.600	0.000	0.583	0.500	0.100	0.600	0.600	0.600	0.375	0.444	0.600	0.313	0.600
14	0.600	0.600	0.000	0.667	0.600	0.100	0.700	0.600	0.600	0.375	0.444	0.700	0.375	0.700
15	0.600	0.600	**0.000**	0.667	0.600	**0.000**	0.600	0.600	0.600	0.375	0.444	0.600	0.375	0.600

Table 5-17: Variation in W_S and W_P Values with Faulty Voltage, Torque and Speed Sensors

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.400	0.400	0.400	0.500	0.500	0.500	0.500	0.400	0.400	0.500	0.500	0.500	0.500	0.500
2	0.500	0.500	0.400	0.583	0.500	0.500	0.500	0.500	0.500	0.563	0.500	0.500	0.500	0.500
3	0.400	0.400	0.400	0.583	0.400	0.500	0.500	0.400	0.400	0.500	0.444	0.500	0.500	0.500
4	0.300	0.300	0.400	0.583	0.400	0.400	0.500	0.300	0.300	0.438	0.389	0.500	0.438	0.500
5	0.400	0.400	0.400	0.583	0.400	0.400	0.600	0.400	0.400	0.500	0.444	0.600	0.500	0.600
6	0.400	0.400	0.300	0.500	0.400	0.400	0.600	0.400	0.400	0.438	0.444	0.600	0.500	0.600
7	0.400	0.400	0.400	0.500	0.500	0.400	0.600	0.400	0.400	0.500	0.500	0.600	0.500	0.600
8	0.400	0.400	0.300	0.500	0.500	0.300	0.600	0.400	0.400	0.438	0.444	0.600	0.438	0.600
9	0.400	0.400	0.200	0.500	0.500	0.300	0.500	0.400	0.400	0.375	0.389	0.500	0.375	0.500
10	0.400	0.400	0.200	0.417	0.400	0.300	0.500	0.400	0.400	0.375	0.333	0.500	0.375	0.500
11	0.400	0.400	0.200	0.333	0.400	0.200	0.500	0.400	0.400	0.375	0.278	0.500	0.313	0.500
12	0.400	0.400	0.200	0.417	0.400	0.200	0.600	0.400	0.400	0.375	0.333	0.600	0.375	0.600
13	0.400	0.400	0.200	0.417	0.300	0.100	0.600	0.400	0.400	0.375	0.333	0.600	0.313	0.600
14	0.400	0.400	0.200	0.333	0.200	0.100	0.500	0.400	0.400	0.375	0.333	0.500	0.250	0.500
15	0.400	0.400	**0.200**	0.333	**0.200**	**0.000**	0.400	0.400	0.400	0.375	0.333	0.400	0.250	0.400

that the algorithm reliably isolates the potentially faulty sensors as long as the number of faulty sensors was less than the number of operational/normal sensors in the system.

Now consider as an example of a fault in the *V->I* process. Such a fault may be the result of a change in the resistance of the motor windings (in other words, confirmation of this fault could indicate the existence of a potential winding fault). Such a fault can cause a complete change in the operational characteristics of the motor. For instance, for the same level supply voltage (V_S) and control parameter (*DC* and *FR*) conditions, the corresponding values of all the variables (Figure 5-15) are now decreased.

Suppose the sensors now indicate 0.2 V for the voltage (same as before) but 0.9 A, 0.6 Nm, 350 rpm and 72 dB for current, torque, speed and acoustic noise respectively. It can be seen that although the readings for four of the sensors deviates from their normal or expected values, the SPFDI algorithm is still able to correctly distinguish the cause of deviations as a fault in the *V->I* process as evident from the minimum W_P value for that process when compared to all the other W_S and W_P values (Table 5-18).

Consider another example where the load torque acting on the motor suddenly increases. This implies that for a given motor torque, the motor speed will decrease[23]. Thus the relation *T->S* as encoded in the CPT of node *S*, no longer holds true. The sudden increase in the load torque can therefore be thought of as a fault in the *T->S* process (or conversely if a *T->S* process fault is determined from the algorithm results, the operator may use the $W_{T->S}$ value and interpret it to be the effect of a change in the load torque). The reduction in speed also causes a reduction in the acoustic noise generated (primarily due to reduced windage effects). Thus both the speed and acoustic noise sensors would indicate readings lower than their expected values under regular operating conditions.

Suppose all the sensors indicate the same values as before, with the speed value now at 350 rpm and a noise level reading of 71 dB.

[23] Assuming the absence of any other control action to maintain the speed at its command value

Table 5-18: Variation in W_S and W_P Values with a *V->I* Process Fault

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.600	0.600	0.600	0.500	0.500	0.500	0.500	0.600	0.600	0.500	0.500	0.500	0.500	0.500
2	0.500	0.500	0.600	0.417	0.500	0.500	0.500	0.500	0.500	0.438	0.500	0.500	0.500	0.500
3	0.400	0.400	0.600	0.417	0.400	0.500	0.500	0.400	0.400	0.375	0.444	0.500	0.500	0.500
4	0.300	0.300	0.600	0.417	0.400	0.400	0.500	0.300	0.300	0.313	0.389	0.500	0.438	0.500
5	0.200	0.200	0.600	0.417	0.400	0.400	0.400	0.200	0.200	0.250	0.333	0.400	0.375	0.400
6	0.200	0.200	0.500	0.333	0.400	0.400	0.400	0.200	0.200	0.188	0.333	0.400	0.375	0.400
7	0.200	0.200	0.400	0.333	0.300	0.400	0.400	0.200	0.200	0.125	0.278	0.400	0.375	0.400
8	0.200	0.200	0.300	0.333	0.300	0.300	0.400	0.200	0.200	0.063	0.222	0.400	0.313	0.400
9	0.200	0.200	0.200	0.333	0.300	0.300	0.300	0.200	0.200	0.000	0.167	0.300	0.250	0.300
10	0.200	0.200	0.200	0.417	0.400	0.300	0.300	0.200	0.200	0.000	0.222	0.300	0.250	0.300
11	0.200	0.200	0.200	0.500	0.400	0.400	0.300	0.200	0.200	0.000	0.278	0.300	0.313	0.300
12	0.200	0.200	0.200	0.583	0.400	0.400	0.400	0.200	0.200	0.000	0.333	0.400	0.375	0.400
13	0.200	0.200	0.200	0.583	0.500	0.500	0.400	0.200	0.200	0.000	0.333	0.400	0.438	0.400
14	0.200	0.200	0.200	0.667	0.600	0.500	0.500	0.200	0.200	0.000	0.333	0.500	0.500	0.500
15	0.200	0.200	0.200	0.667	0.600	0.600	0.600	0.200	0.200	**0.000**	0.333	0.600	0.500	0.600

Table 5-19: Variation in W_S and W_P Values with a *T->S* Process Fault

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.600	0.600	0.600	0.500	0.500	0.500	0.500	0.600	0.600	0.500	0.500	0.500	0.500	0.500
2	0.700	0.700	0.600	0.571	0.500	0.500	0.500	0.700	0.700	0.563	0.500	0.500	0.500	0.500
3	0.800	0.800	0.600	0.571	0.600	0.500	0.500	0.800	0.800	0.625	0.556	0.500	0.500	0.500
4	0.700	0.700	0.600	0.571	0.600	0.400	0.500	0.700	0.700	0.563	0.500	0.500	0.438	0.500
5	0.600	0.600	0.600	0.571	0.600	0.400	0.400	0.600	0.600	0.500	0.444	0.400	0.375	0.400
6	0.600	0.600	0.700	0.643	0.600	0.400	0.400	0.600	0.600	0.563	0.444	0.400	0.375	0.400
7	0.600	0.600	0.800	0.643	0.700	0.400	0.400	0.600	0.600	0.625	0.500	0.400	0.375	0.400
8	0.600	0.600	0.700	0.643	0.700	0.300	0.400	0.600	0.600	0.563	0.444	0.400	0.313	0.400
9	0.600	0.600	0.600	0.643	0.700	0.300	0.300	0.600	0.600	0.500	0.389	0.300	0.250	0.300
10	0.600	0.600	0.600	0.714	0.800	0.300	0.300	0.600	0.600	0.500	0.444	0.300	0.250	0.300
11	0.600	0.600	0.600	0.643	0.800	0.200	0.300	0.600	0.600	0.500	0.389	0.300	0.188	0.300
12	0.600	0.600	0.600	0.571	0.800	0.200	0.200	0.600	0.600	0.500	0.333	0.200	0.125	0.200
13	0.600	0.600	0.600	0.571	0.700	0.100	0.200	0.600	0.600	0.500	0.333	0.200	0.063	0.200
14	0.600	0.600	0.600	0.500	0.600	0.100	0.100	0.600	0.600	0.500	0.333	0.100	0.000	0.100
15	0.600	0.600	0.600	0.571	0.600	0.200	0.200	0.600	0.600	0.500	0.333	0.200	**0.000**	0.200

If these values are considered individually, then they would be construed as being indicative of a fault in both the speed and the noise sensor. However, by correlating these values with the values indicated by the other sensors in the Bayesian network, the algorithm is correctly able to isolate the cause of variation as being a change in the *T*->*S* process rather than dual sensor faults or other process faults (e.g. *I*->*N* or *S*->*N*). Table 5-19 shows the changes in the belief values for the different sensors and processes in the network in this case.

If, for a commanded value of *DC* and *FR*, the voltage is not what it is supposed to be, but the values for all the other variables are consistent with the measured value of *V*, then it could mean that there is potentially a fault in the *DC*->*V*<-*FR* processes (process fault). This may be due to some issue with the power amplifiers that convert the PWM commands to the desired voltage. Alternatively, it could also mean that there is no issue with the amplifiers; the sensors may be indicating abnormal values because a wrong command was received by the amplifiers (e.g. due to an inconsistent PWM signal as a result of a fault in the line relaying the user commands from the main controller/PC to the amplifiers) and the motor is not actually operating at the conditions it was supposed to (this would be akin to a 'sensor' fault). Considering the case of a *DC*->*V* process fault[24], suppose the value of *DC* is now 5% but all the other sensors continue to indicate the same values shown in Figure 5-15 (ideally all these values should now be higher because of the higher supply voltage). The changes in the W_S and W_P values are shown in Table 5-20.

It is seen that the W_P values for both the *DC*->*V* and *FR*->*V* processes are reduced to zero. In addition, the W_S weights for both the control parameter nodes are also reduced to zero indicating that both are potentially faulty, which is an incorrect result. Thus, as explained earlier, with a deviation in the values corresponding to a root node/process, the algorithm cannot reliably distinguish between sensor and process faults.

[24] Physically, it is not very intuitive to distinguish a deviation in the value of *V* as being caused by a fault in the *DC*->*V* or the *FR*->*V* process considering them separately, since the resultant value of voltage depends on the values of both the PWM duty cycle and its frequency.

Table 5-20: Variation in W_S and W_P Values with a *DC->V* Process Fault

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.400	0.400	0.400	0.500	0.500	0.500	0.500	0.400	0.400	0.500	0.500	0.500	0.500	0.500
2	0.300	0.300	0.400	0.429	0.500	0.500	0.500	0.300	0.300	0.438	0.500	0.500	0.500	0.500
3	0.200	0.200	0.400	0.429	0.400	0.500	0.500	0.200	0.200	0.375	0.444	0.500	0.500	0.500
4	0.100	0.100	0.400	0.429	0.400	0.400	0.500	0.100	0.100	0.313	0.389	0.500	0.438	0.500
5	0.000	0.000	0.400	0.429	0.400	0.400	0.400	0.000	0.000	0.250	0.333	0.400	0.375	0.400
6	0.000	0.000	0.500	0.500	0.400	0.400	0.400	0.000	0.000	0.313	0.333	0.400	0.375	0.400
7	0.000	0.000	0.600	0.500	0.500	0.400	0.400	0.000	0.000	0.375	0.389	0.400	0.375	0.400
8	0.000	0.000	0.700	0.500	0.500	0.500	0.400	0.000	0.000	0.438	0.444	0.400	0.438	0.400
9	0.000	0.000	0.800	0.500	0.500	0.500	0.500	0.000	0.000	0.500	0.500	0.500	0.500	0.500
10	0.000	0.000	0.800	0.571	0.600	0.500	0.500	0.000	0.000	0.500	0.556	0.500	0.500	0.500
11	0.000	0.000	0.800	0.643	0.600	0.600	0.500	0.000	0.000	0.500	0.611	0.500	0.563	0.500
12	0.000	0.000	0.800	0.714	0.600	0.600	0.600	0.000	0.000	0.500	0.667	0.600	0.625	0.600
13	0.000	0.000	0.800	0.714	0.700	0.700	0.600	0.000	0.000	0.500	0.667	0.600	0.688	0.600
14	0.000	0.000	0.800	0.786	0.800	0.700	0.700	0.000	0.000	0.500	0.667	0.700	0.750	0.700
15	**0.000**	**0.000**	0.800	0.857	0.800	0.800	0.800	**0.000**	**0.000**	0.500	0.667	0.800	0.750	0.800

Table 5-21: Variation in W_S and W_P Values with a Faulty *N* Sensor

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
1	0.600	0.600	0.600	0.500	0.500	0.500	0.500	0.600	0.600	0.500	0.500	0.500	0.500	0.500
2	0.700	0.700	0.600	0.571	0.500	0.500	0.500	0.700	0.700	0.563	0.500	0.500	0.500	0.500
3	0.800	0.800	0.600	0.571	0.600	0.500	0.500	0.800	0.800	0.625	0.556	0.500	0.500	0.500
4	0.900	0.900	0.600	0.571	0.600	0.600	0.500	0.900	0.900	0.688	0.611	0.500	0.563	0.500
5	0.800	0.800	0.600	0.571	0.600	0.600	0.400	0.800	0.800	0.625	0.556	0.400	0.500	0.400
6	0.800	0.800	0.700	0.643	0.600	0.600	0.400	0.800	0.800	0.688	0.556	0.400	0.500	0.400
7	0.800	0.800	0.800	0.643	0.700	0.600	0.400	0.800	0.800	0.750	0.611	0.400	0.500	0.400
8	0.800	0.800	0.900	0.643	0.700	0.700	0.400	0.800	0.800	0.813	0.667	0.400	0.563	0.400
9	0.800	0.800	0.800	0.643	0.700	0.700	0.300	0.800	0.800	0.750	0.611	0.300	0.500	0.300
10	0.800	0.800	0.800	0.714	0.800	0.700	0.300	0.800	0.800	0.750	0.667	0.300	0.500	0.300
11	0.800	0.800	0.800	0.786	0.800	0.800	0.300	0.800	0.800	0.750	0.722	0.300	0.563	0.300
12	0.800	0.800	0.800	0.714	0.800	0.800	0.200	0.800	0.800	0.750	0.667	0.200	0.500	0.200
13	0.800	0.800	0.800	0.714	0.900	0.900	0.200	0.800	0.800	0.750	0.667	0.200	0.563	0.200
14	0.800	0.800	0.800	0.643	0.800	0.900	0.100	0.800	0.800	0.750	0.667	0.100	0.500	0.100
15	0.800	0.800	0.800	0.571	0.800	0.800	**0.000**	0.800	0.800	0.750	0.667	**0.000**	0.500	**0.000**

In the case of a deviation in the readings from a sensor corresponding to a leaf node, the algorithm is again not completely reliable as explained earlier. For example,

suppose the acoustic noise sensor is faulty and indicates a reading of 70 dB for the N node instead of 74 dB, whereas all the other sensors continue to indicate the same values as before. It can be seen from Table 5-21, the W_S weights for the leaf node N and the W_P weights for both the S->N and I->N processes are reduced to zero, resulting in an ambiguous result.

To accommodate the two cases mentioned above, redundant nodes (see Section 5.5.4) are added to root and leaf level nodes. The list of additional instantiations obtained by adding such redundant nodes to the top is shown in Table 5-22. Note that even with the addition of redundant nodes $Col_{DC->V} \equiv Col_{FR->V}$. The variation in W_S and W_P values with different types of sensor and process faults is shown in Table 5-23 to Table 5-28.

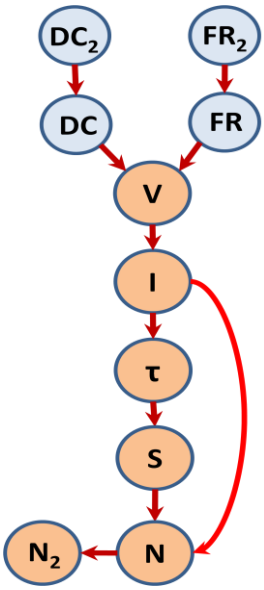


Figure 5-18: Network Augmented with Redundant Sensors

In each case, it is observed that the SPFDI algorithm is able detect and isolate the exact source of faults. Due to the identical columns for DC->V and FR->V, when a fault is assumed in the DC->V process, the W_P values for both the processes are reduced to zero. This can be still considered a meaningful result, since it confirms that the variation in the voltage sensor reading is due to a fault in the voltage amplification process and not due to a faulty voltage sensor. As explained in Section 5.5.4, the (ideal) processes DC_2->DC, FR_2->FR and N_2->N are not real processes and their W_P values are always set to 1 throughout the fault detection and isolation cycle.

Table 5-22 : Instantiation Table for the Testbed with Redundant Nodes

	Nodes										Processes									
Step	**DC_2**	**DC**	**FR_2**	**FR**	**V**	**I**	**T**	**S**	**N**	**N_2**	**DC_2-DC**	**FR_2-FR**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**	**N-N_2**
1																				
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				
11																				
12																				
13																				
14																				
15																				
16																				
17																				
18																				
19																				
20																				
21																				
22																				
23																				
24																				
25																				
26																				
27																				
28																				
29																				

Node instantiated as evidence Node of interest Process of Interest

Table 5-23 : Variation in W_S and W_P Values with No Sensor or Process Faults

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	$N-N_2$
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
1	0.571	0.500	0.571	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
2	0.571	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
3	0.643	0.643	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.542	0.542	0.500	0.500	0.500	0.500	0.500	1.000
4	0.714	0.714	0.571	0.571	0.571	0.545	0.500	0.500	0.500	0.500	1.000	1.000	0.583	0.583	0.533	0.500	0.500	0.500	0.500	1.000
5	0.786	0.786	0.571	0.571	0.571	0.545	0.571	0.500	0.500	0.500	1.000	1.000	0.625	0.625	0.567	0.525	0.500	0.500	0.500	1.000
6	0.857	0.857	0.571	0.571	0.571	0.545	0.571	0.571	0.500	0.500	1.000	1.000	0.667	0.667	0.600	0.550	0.500	0.529	0.500	1.000
7	0.929	0.929	0.571	0.571	0.571	0.545	0.571	0.571	0.571	0.500	1.000	1.000	0.708	0.708	0.633	0.575	0.542	0.559	0.542	1.000
8	1.000	1.000	0.571	0.571	0.571	0.545	0.571	0.571	0.571	0.571	1.000	1.000	0.750	0.750	0.667	0.600	0.583	0.588	0.583	1.000
9	1.000	1.000	0.643	0.643	0.643	0.545	0.571	0.571	0.571	0.571	1.000	1.000	0.792	0.792	0.667	0.600	0.583	0.588	0.583	1.000
10	1.000	1.000	0.714	0.714	0.643	0.591	0.571	0.571	0.571	0.571	1.000	1.000	0.833	0.833	0.700	0.600	0.583	0.588	0.583	1.000
11	1.000	1.000	0.786	0.786	0.643	0.591	0.643	0.571	0.571	0.571	1.000	1.000	0.875	0.875	0.733	0.625	0.583	0.588	0.583	1.000
12	1.000	1.000	0.857	0.857	0.643	0.591	0.643	0.643	0.571	0.571	1.000	1.000	0.917	0.917	0.767	0.650	0.583	0.618	0.583	1.000
13	1.000	1.000	0.929	0.929	0.643	0.591	0.643	0.643	0.643	0.571	1.000	1.000	0.958	0.958	0.800	0.675	0.625	0.647	0.625	1.000
14	1.000	1.000	1.000	1.000	0.643	0.591	0.643	0.643	0.643	0.643	1.000	1.000	1.000	1.000	0.833	0.700	0.667	0.676	0.667	1.000
15	1.000	1.000	1.000	1.000	0.714	0.636	0.643	0.643	0.643	0.643	1.000	1.000	1.000	1.000	0.867	0.700	0.667	0.676	0.667	1.000
16	1.000	1.000	1.000	1.000	0.786	0.636	0.714	0.643	0.643	0.643	1.000	1.000	1.000	1.000	0.900	0.725	0.667	0.676	0.667	1.000
17	1.000	1.000	1.000	1.000	0.857	0.636	0.714	0.714	0.643	0.643	1.000	1.000	1.000	1.000	0.933	0.750	0.667	0.706	0.667	1.000
18	1.000	1.000	1.000	1.000	0.929	0.636	0.714	0.714	0.714	0.643	1.000	1.000	1.000	1.000	0.967	0.775	0.708	0.735	0.708	1.000
19	1.000	1.000	1.000	1.000	1.000	0.636	0.714	0.714	0.714	0.714	1.000	1.000	1.000	1.000	1.000	0.800	0.750	0.765	0.750	1.000
20	1.000	1.000	1.000	1.000	1.000	0.682	0.786	0.714	0.714	0.714	1.000	1.000	1.000	1.000	1.000	0.825	0.750	0.765	0.750	1.000
21	1.000	1.000	1.000	1.000	1.000	0.727	0.786	0.786	0.714	0.714	1.000	1.000	1.000	1.000	1.000	0.850	0.750	0.794	0.750	1.000
22	1.000	1.000	1.000	1.000	1.000	0.773	0.786	0.786	0.786	0.714	1.000	1.000	1.000	1.000	1.000	0.875	0.792	0.824	0.792	1.000
23	1.000	1.000	1.000	1.000	1.000	0.818	0.786	0.786	0.786	0.786	1.000	1.000	1.000	1.000	1.000	0.900	0.833	0.853	0.833	1.000
24	1.000	1.000	1.000	1.000	1.000	0.818	0.857	0.857	0.786	0.786	1.000	1.000	1.000	1.000	1.000	0.900	0.833	0.882	0.833	1.000
25	1.000	1.000	1.000	1.000	1.000	0.864	0.929	0.857	0.857	0.786	1.000	1.000	1.000	1.000	1.000	0.925	0.875	0.912	0.875	1.000
26	1.000	1.000	1.000	1.000	1.000	0.909	1.000	0.857	0.857	0.857	1.000	1.000	1.000	1.000	1.000	0.950	0.917	0.941	0.917	1.000
27	1.000	1.000	1.000	1.000	1.000	0.955	1.000	0.929	0.929	0.857	1.000	1.000	1.000	1.000	1.000	0.975	0.958	0.971	0.958	1.000
28	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.929	0.929	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
29	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

Table 5-24: Variation in W_S and W_P Values with a Faulty Voltage Sensor

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
1	0.571	0.500	0.571	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
2	0.571	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
3	0.500	0.500	0.571	0.571	0.429	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.458	0.458	0.500	0.500	0.500	0.500	0.500	1.000
4	0.571	0.571	0.571	0.571	0.429	0.545	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.533	0.500	0.500	0.500	0.500	1.000
5	0.643	0.643	0.571	0.571	0.429	0.545	0.571	0.500	0.500	0.500	1.000	1.000	0.542	0.542	0.567	0.525	0.500	0.500	0.500	1.000
6	0.714	0.714	0.571	0.571	0.429	0.545	0.571	0.571	0.500	0.500	1.000	1.000	0.583	0.583	0.600	0.550	0.500	0.529	0.500	1.000
7	0.786	0.786	0.571	0.571	0.429	0.545	0.571	0.571	0.571	0.500	1.000	1.000	0.625	0.625	0.633	0.575	0.542	0.559	0.542	1.000
8	0.857	0.857	0.571	0.571	0.429	0.545	0.571	0.571	0.571	0.571	1.000	1.000	0.667	0.667	0.667	0.600	0.583	0.588	0.583	1.000
9	0.857	0.857	0.500	0.500	0.357	0.545	0.571	0.571	0.571	0.571	1.000	1.000	0.625	0.625	0.667	0.600	0.583	0.588	0.583	1.000
10	0.857	0.857	0.571	0.571	0.357	0.591	0.571	0.571	0.571	0.571	1.000	1.000	0.667	0.667	0.700	0.600	0.583	0.588	0.583	1.000
11	0.857	0.857	0.643	0.643	0.357	0.591	0.643	0.571	0.571	0.571	1.000	1.000	0.708	0.708	0.733	0.625	0.583	0.588	0.583	1.000
12	0.857	0.857	0.714	0.714	0.357	0.591	0.643	0.643	0.571	0.571	1.000	1.000	0.750	0.750	0.767	0.650	0.583	0.618	0.583	1.000
13	0.857	0.857	0.786	0.786	0.357	0.591	0.643	0.643	0.643	0.571	1.000	1.000	0.792	0.792	0.800	0.675	0.625	0.647	0.625	1.000
14	0.857	0.857	0.857	0.857	0.357	0.591	0.643	0.643	0.643	0.643	1.000	1.000	0.833	0.833	0.833	0.700	0.667	0.676	0.667	1.000
15	0.857	0.857	0.857	0.857	0.286	0.545	0.643	0.643	0.643	0.643	1.000	1.000	0.833	0.833	0.800	0.700	0.667	0.676	0.667	1.000
16	0.857	0.857	0.857	0.857	0.214	0.545	0.571	0.643	0.643	0.643	1.000	1.000	0.833	0.833	0.767	0.675	0.667	0.676	0.667	1.000
17	0.857	0.857	0.857	0.857	0.143	0.545	0.571	0.571	0.643	0.643	1.000	1.000	0.833	0.833	0.733	0.650	0.667	0.647	0.667	1.000
18	0.857	0.857	0.857	0.857	0.071	0.545	0.571	0.571	0.571	0.643	1.000	1.000	0.833	0.833	0.700	0.625	0.625	0.618	0.625	1.000
19	0.857	0.857	0.857	0.857	0.000	0.545	0.571	0.571	0.571	0.571	1.000	1.000	0.833	0.833	0.667	0.600	0.583	0.588	0.583	1.000
20	0.857	0.857	0.857	0.857	0.000	0.591	0.643	0.571	0.571	0.571	1.000	1.000	0.833	0.833	0.667	0.625	0.583	0.588	0.583	1.000
21	0.857	0.857	0.857	0.857	0.000	0.636	0.643	0.643	0.571	0.571	1.000	1.000	0.833	0.833	0.667	0.650	0.583	0.618	0.583	1.000
22	0.857	0.857	0.857	0.857	0.000	0.682	0.643	0.643	0.643	0.571	1.000	1.000	0.833	0.833	0.667	0.675	0.625	0.647	0.625	1.000
23	0.857	0.857	0.857	0.857	0.000	0.727	0.643	0.643	0.643	0.643	1.000	1.000	0.833	0.833	0.667	0.700	0.667	0.676	0.667	1.000
24	0.857	0.857	0.857	0.857	0.000	0.727	0.714	0.714	0.643	0.643	1.000	1.000	0.833	0.833	0.667	0.700	0.667	0.706	0.667	1.000
25	0.857	0.857	0.857	0.857	0.000	0.773	0.786	0.714	0.714	0.643	1.000	1.000	0.833	0.833	0.667	0.725	0.708	0.735	0.708	1.000
26	0.857	0.857	0.857	0.857	0.000	0.818	0.857	0.714	0.714	0.714	1.000	1.000	0.833	0.833	0.667	0.750	0.750	0.765	0.750	1.000
27	0.857	0.857	0.857	0.857	0.000	0.864	0.857	0.786	0.786	0.714	1.000	1.000	0.833	0.833	0.667	0.775	0.792	0.794	0.792	1.000
28	0.857	0.857	0.857	0.857	0.000	0.909	0.857	0.857	0.786	0.786	1.000	1.000	0.833	0.833	0.667	0.800	0.833	0.824	0.833	1.000
29	0.857	0.857	0.857	0.857	**0.000**	0.909	0.857	0.857	0.857	0.857	1.000	1.000	0.833	0.833	0.667	0.800	0.833	0.824	0.833	1.000

Table 5-25: Variation in W_S and W_P Values with Faulty Voltage and Speed Sensors

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
1	0.571	0.500	0.571	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
2	0.571	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
3	0.500	0.500	0.571	0.571	0.429	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.458	0.458	0.500	0.500	0.500	0.500	0.500	1.000
4	0.571	0.571	0.571	0.571	0.429	0.545	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.533	0.500	0.500	0.500	0.500	1.000
5	0.643	0.643	0.571	0.571	0.429	0.545	0.571	0.500	0.500	0.500	1.000	1.000	0.542	0.542	0.567	0.525	0.500	0.500	0.500	1.000
6	0.571	0.571	0.571	0.571	0.429	0.545	0.571	0.429	0.500	0.500	1.000	1.000	0.500	0.500	0.533	0.500	0.500	0.471	0.500	1.000
7	0.643	0.643	0.571	0.571	0.429	0.545	0.571	0.429	0.571	0.500	1.000	1.000	0.542	0.542	0.567	0.525	0.542	0.500	0.542	1.000
8	0.714	0.714	0.571	0.571	0.429	0.545	0.571	0.429	0.571	0.571	1.000	1.000	0.583	0.583	0.600	0.550	0.583	0.529	0.583	1.000
9	0.714	0.714	0.500	0.500	0.357	0.545	0.571	0.429	0.571	0.571	1.000	1.000	0.542	0.542	0.600	0.550	0.583	0.529	0.583	1.000
10	0.714	0.714	0.571	0.571	0.357	0.591	0.571	0.429	0.571	0.571	1.000	1.000	0.583	0.583	0.633	0.550	0.583	0.529	0.583	1.000
11	0.714	0.714	0.643	0.643	0.357	0.591	0.643	0.429	0.571	0.571	1.000	1.000	0.625	0.625	0.667	0.575	0.583	0.529	0.583	1.000
12	0.714	0.714	0.571	0.571	0.357	0.591	0.643	0.357	0.571	0.571	1.000	1.000	0.583	0.583	0.633	0.550	0.583	0.500	0.583	1.000
13	0.714	0.714	0.643	0.643	0.357	0.591	0.643	0.357	0.643	0.571	1.000	1.000	0.625	0.625	0.667	0.575	0.625	0.529	0.625	1.000
14	0.714	0.714	0.714	0.714	0.357	0.591	0.643	0.357	0.643	0.643	1.000	1.000	0.667	0.667	0.700	0.600	0.667	0.559	0.667	1.000
15	0.714	0.714	0.714	0.714	0.286	0.545	0.643	0.357	0.643	0.643	1.000	1.000	0.667	0.667	0.667	0.600	0.667	0.559	0.667	1.000
16	0.714	0.714	0.714	0.714	0.214	0.545	0.571	0.357	0.643	0.643	1.000	1.000	0.667	0.667	0.633	0.575	0.667	0.559	0.667	1.000
17	0.714	0.714	0.714	0.714	0.143	0.545	0.571	0.286	0.643	0.643	1.000	1.000	0.667	0.667	0.600	0.550	0.667	0.529	0.667	1.000
18	0.714	0.714	0.714	0.714	0.071	0.545	0.571	0.286	0.571	0.643	1.000	1.000	0.667	0.667	0.567	0.525	0.625	0.500	0.625	1.000
19	0.714	0.714	0.714	0.714	0.000	0.545	0.571	0.286	0.571	0.571	1.000	1.000	0.667	0.667	0.533	0.500	0.583	0.471	0.583	1.000
20	0.714	0.714	0.714	0.714	0.000	0.591	0.643	0.286	0.571	0.571	1.000	1.000	0.667	0.667	0.533	0.525	0.583	0.471	0.583	1.000
21	0.714	0.714	0.714	0.714	0.000	0.545	0.643	0.214	0.571	0.571	1.000	1.000	0.667	0.667	0.533	0.500	0.583	0.441	0.583	1.000
22	0.714	0.714	0.714	0.714	0.000	0.591	0.643	0.214	0.643	0.571	1.000	1.000	0.667	0.667	0.533	0.525	0.625	0.471	0.625	1.000
23	0.714	0.714	0.714	0.714	0.000	0.636	0.643	0.214	0.643	0.643	1.000	1.000	0.667	0.667	0.533	0.550	0.667	0.500	0.667	1.000
24	0.714	0.714	0.714	0.714	0.000	0.636	0.571	0.143	0.643	0.643	1.000	1.000	0.667	0.667	0.533	0.550	0.667	0.471	0.667	1.000
25	0.714	0.714	0.714	0.714	0.000	0.682	0.643	0.143	0.714	0.643	1.000	1.000	0.667	0.667	0.533	0.575	0.708	0.500	0.708	1.000
26	0.714	0.714	0.714	0.714	0.000	0.727	0.714	0.143	0.714	0.714	1.000	1.000	0.667	0.667	0.533	0.600	0.750	0.529	0.750	1.000
27	0.714	0.714	0.714	0.714	0.000	0.682	0.714	0.071	0.643	0.714	1.000	1.000	0.667	0.667	0.533	0.575	0.708	0.500	0.708	1.000
28	0.714	0.714	0.714	0.714	0.000	0.636	0.714	0.000	0.643	0.643	1.000	1.000	0.667	0.667	0.533	0.550	0.667	0.471	0.667	1.000
29	0.714	0.714	0.714	0.714	**0.000**	0.636	0.714	**0.000**	0.714	0.714	1.000	1.000	0.667	0.667	0.533	0.550	0.667	0.471	0.667	1.000

Table 5-26: Variation in W_S and W_P Values with a *V->I* Process Fault

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	$N-N_2$
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
1	0.571	0.500	0.571	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
2	0.571	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
3	0.643	0.643	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.542	0.542	0.500	0.500	0.500	0.500	0.500	1.000
4	0.571	0.571	0.571	0.571	0.571	0.455	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.467	0.500	0.500	0.500	0.500	1.000
5	0.500	0.500	0.571	0.571	0.571	0.455	0.429	0.500	0.500	0.500	1.000	1.000	0.458	0.458	0.433	0.475	0.500	0.500	0.500	1.000
6	0.429	0.429	0.571	0.571	0.571	0.455	0.429	0.429	0.500	0.500	1.000	1.000	0.417	0.417	0.400	0.450	0.500	0.471	0.500	1.000
7	0.357	0.357	0.571	0.571	0.571	0.455	0.429	0.429	0.429	0.500	1.000	1.000	0.375	0.375	0.367	0.425	0.458	0.441	0.458	1.000
8	0.286	0.286	0.571	0.571	0.571	0.455	0.429	0.429	0.429	0.429	1.000	1.000	0.333	0.333	0.333	0.400	0.417	0.412	0.417	1.000
9	0.286	0.286	0.643	0.643	0.643	0.455	0.429	0.429	0.429	0.429	1.000	1.000	0.375	0.375	0.333	0.400	0.417	0.412	0.417	1.000
10	0.286	0.286	0.571	0.571	0.643	0.409	0.429	0.429	0.429	0.429	1.000	1.000	0.333	0.333	0.300	0.400	0.417	0.412	0.417	1.000
11	0.286	0.286	0.500	0.500	0.643	0.409	0.357	0.429	0.429	0.429	1.000	1.000	0.292	0.292	0.267	0.375	0.417	0.412	0.417	1.000
12	0.286	0.286	0.429	0.429	0.643	0.409	0.357	0.357	0.429	0.429	1.000	1.000	0.250	0.250	0.233	0.350	0.417	0.382	0.417	1.000
13	0.286	0.286	0.357	0.357	0.643	0.409	0.357	0.357	0.357	0.429	1.000	1.000	0.208	0.208	0.200	0.325	0.375	0.353	0.375	1.000
14	0.286	0.286	0.286	0.286	0.643	0.409	0.357	0.357	0.357	0.357	1.000	1.000	0.167	0.167	0.167	0.300	0.333	0.324	0.333	1.000
15	0.286	0.286	0.286	0.286	0.571	0.364	0.357	0.357	0.357	0.357	1.000	1.000	0.167	0.167	0.133	0.300	0.333	0.324	0.333	1.000
16	0.286	0.286	0.286	0.286	0.500	0.364	0.286	0.357	0.357	0.357	1.000	1.000	0.167	0.167	0.100	0.275	0.333	0.324	0.333	1.000
17	0.286	0.286	0.286	0.286	0.429	0.364	0.286	0.286	0.357	0.357	1.000	1.000	0.167	0.167	0.067	0.250	0.333	0.294	0.333	1.000
18	0.286	0.286	0.286	0.286	0.357	0.364	0.286	0.286	0.286	0.357	1.000	1.000	0.167	0.167	0.033	0.225	0.292	0.265	0.292	1.000
19	0.286	0.286	0.286	0.286	0.286	0.364	0.286	0.286	0.286	0.286	1.000	1.000	0.167	0.167	0.000	0.200	0.250	0.235	0.250	1.000
20	0.286	0.286	0.286	0.286	0.286	0.409	0.357	0.286	0.286	0.286	1.000	1.000	0.167	0.167	0.000	0.225	0.250	0.235	0.250	1.000
21	0.286	0.286	0.286	0.286	0.286	0.455	0.357	0.357	0.286	0.286	1.000	1.000	0.167	0.167	0.000	0.250	0.250	0.265	0.250	1.000
22	0.286	0.286	0.286	0.286	0.286	0.500	0.357	0.357	0.357	0.286	1.000	1.000	0.167	0.167	0.000	0.275	0.292	0.294	0.292	1.000
23	0.286	0.286	0.286	0.286	0.286	0.455	0.357	0.357	0.357	0.214	1.000	1.000	0.167	0.167	0.000	0.250	0.250	0.265	0.250	1.000
24	0.286	0.286	0.286	0.286	0.286	0.455	0.429	0.429	0.357	0.214	1.000	1.000	0.167	0.167	0.000	0.250	0.250	0.294	0.250	1.000
25	0.286	0.286	0.286	0.286	0.286	0.500	0.500	0.429	0.429	0.214	1.000	1.000	0.167	0.167	0.000	0.275	0.292	0.324	0.292	1.000
26	0.286	0.286	0.286	0.286	0.286	0.545	0.571	0.429	0.429	0.286	1.000	1.000	0.167	0.167	0.000	0.300	0.333	0.353	0.333	1.000
27	0.286	0.286	0.286	0.286	0.286	0.591	0.571	0.500	0.500	0.286	1.000	1.000	0.167	0.167	0.000	0.325	0.375	0.382	0.375	1.000
28	0.286	0.286	0.286	0.286	0.286	0.636	0.571	0.571	0.500	0.357	1.000	1.000	0.167	0.167	0.000	0.350	0.417	0.412	0.417	1.000
29	0.286	0.286	0.286	0.286	0.286	0.636	0.571	0.571	0.571	0.429	1.000	1.000	0.167	0.167	**0.000**	0.350	0.417	0.412	0.417	1.000

Table 5-27: Variation in W_S and W_P Values with a *DC->V* Process Fault

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	$N-N_2$
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
1	0.571	0.500	0.571	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
2	0.571	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
3	0.500	0.500	0.571	0.571	0.429	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.458	0.458	0.500	0.500	0.500	0.500	0.500	1.000
4	0.429	0.429	0.571	0.571	0.429	0.455	0.500	0.500	0.500	0.500	1.000	1.000	0.417	0.417	0.467	0.500	0.500	0.500	0.500	1.000
5	0.357	0.357	0.571	0.571	0.429	0.455	0.429	0.500	0.500	0.500	1.000	1.000	0.375	0.375	0.433	0.475	0.500	0.500	0.500	1.000
6	0.286	0.286	0.571	0.571	0.429	0.455	0.429	0.429	0.500	0.500	1.000	1.000	0.333	0.333	0.400	0.450	0.500	0.471	0.500	1.000
7	0.214	0.214	0.571	0.571	0.429	0.455	0.429	0.429	0.429	0.500	1.000	1.000	0.292	0.292	0.367	0.425	0.458	0.441	0.458	1.000
8	0.143	0.143	0.571	0.571	0.429	0.455	0.429	0.429	0.429	0.429	1.000	1.000	0.250	0.250	0.333	0.400	0.417	0.412	0.417	1.000
9	0.143	0.143	0.500	0.500	0.357	0.455	0.429	0.429	0.429	0.429	1.000	1.000	0.208	0.208	0.333	0.400	0.417	0.412	0.417	1.000
10	0.143	0.143	0.429	0.429	0.357	0.409	0.429	0.429	0.429	0.429	1.000	1.000	0.167	0.167	0.300	0.400	0.417	0.412	0.417	1.000
11	0.143	0.143	0.357	0.357	0.357	0.409	0.357	0.429	0.429	0.429	1.000	1.000	0.125	0.125	0.267	0.375	0.417	0.412	0.417	1.000
12	0.143	0.143	0.286	0.286	0.357	0.409	0.357	0.357	0.429	0.429	1.000	1.000	0.083	0.083	0.233	0.350	0.417	0.382	0.417	1.000
13	0.143	0.143	0.214	0.214	0.357	0.409	0.357	0.357	0.357	0.429	1.000	1.000	0.042	0.042	0.200	0.325	0.375	0.353	0.375	1.000
14	0.143	0.143	0.143	0.143	0.357	0.409	0.357	0.357	0.357	0.357	1.000	1.000	0.000	0.000	0.167	0.300	0.333	0.324	0.333	1.000
15	0.143	0.143	0.143	0.143	0.429	0.455	0.357	0.357	0.357	0.357	1.000	1.000	0.000	0.000	0.200	0.300	0.333	0.324	0.333	1.000
16	0.143	0.143	0.143	0.143	0.500	0.455	0.429	0.357	0.357	0.357	1.000	1.000	0.000	0.000	0.233	0.325	0.333	0.324	0.333	1.000
17	0.143	0.143	0.143	0.143	0.571	0.455	0.429	0.429	0.357	0.357	1.000	1.000	0.000	0.000	0.267	0.350	0.333	0.353	0.333	1.000
18	0.143	0.143	0.143	0.143	0.643	0.455	0.429	0.429	0.429	0.357	1.000	1.000	0.000	0.000	0.300	0.375	0.375	0.382	0.375	1.000
19	0.143	0.143	0.143	0.143	0.714	0.455	0.429	0.429	0.429	0.429	1.000	1.000	0.000	0.000	0.333	0.400	0.417	0.412	0.417	1.000
20	0.143	0.143	0.143	0.143	0.714	0.500	0.500	0.429	0.429	0.429	1.000	1.000	0.000	0.000	0.333	0.425	0.417	0.412	0.417	1.000
21	0.143	0.143	0.143	0.143	0.714	0.545	0.500	0.500	0.429	0.429	1.000	1.000	0.000	0.000	0.333	0.450	0.417	0.441	0.417	1.000
22	0.143	0.143	0.143	0.143	0.714	0.591	0.500	0.500	0.500	0.429	1.000	1.000	0.000	0.000	0.333	0.475	0.458	0.471	0.458	1.000
23	0.143	0.143	0.143	0.143	0.714	0.636	0.500	0.500	0.500	0.500	1.000	1.000	0.000	0.000	0.333	0.500	0.500	0.500	0.500	1.000
24	0.143	0.143	0.143	0.143	0.714	0.636	0.571	0.571	0.500	0.500	1.000	1.000	0.000	0.000	0.333	0.500	0.500	0.529	0.500	1.000
25	0.143	0.143	0.143	0.143	0.714	0.682	0.643	0.571	0.571	0.500	1.000	1.000	0.000	0.000	0.333	0.525	0.542	0.559	0.542	1.000
26	0.143	0.143	0.143	0.143	0.714	0.636	0.571	0.571	0.571	0.429	1.000	1.000	0.000	0.000	0.333	0.500	0.500	0.529	0.500	1.000
27	0.143	0.143	0.143	0.143	0.714	0.591	0.571	0.500	0.500	0.429	1.000	1.000	0.000	0.000	0.333	0.475	0.458	0.500	0.458	1.000
28	0.143	0.143	0.143	0.143	0.714	0.545	0.571	0.429	0.500	0.357	1.000	1.000	0.000	0.000	0.333	0.450	0.417	0.471	0.417	1.000
29	0.143	0.143	0.143	0.143	0.714	0.545	0.571	0.429	0.571	0.429	1.000	1.000	**0.000**	**0.000**	0.333	0.450	0.417	0.471	0.417	1.000

Table 5-28: Variation in W_S and W_P Values with a Faulty Acoustic Noise Sensor

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	$N-N_2$
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
1	0.571	0.500	0.571	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
2	0.571	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
3	0.643	0.643	0.571	0.571	0.571	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.542	0.542	0.500	0.500	0.500	0.500	0.500	1.000
4	0.714	0.714	0.571	0.571	0.571	0.545	0.500	0.500	0.500	0.500	1.000	1.000	0.583	0.583	0.533	0.500	0.500	0.500	0.500	1.000
5	0.786	0.786	0.571	0.571	0.571	0.545	0.571	0.500	0.500	0.500	1.000	1.000	0.625	0.625	0.567	0.525	0.500	0.500	0.500	1.000
6	0.857	0.857	0.571	0.571	0.571	0.545	0.571	0.571	0.500	0.500	1.000	1.000	0.667	0.667	0.600	0.550	0.500	0.529	0.500	1.000
7	0.786	0.786	0.571	0.571	0.571	0.545	0.571	0.571	0.429	0.500	1.000	1.000	0.625	0.625	0.567	0.525	0.458	0.500	0.458	1.000
8	0.857	0.857	0.571	0.571	0.571	0.545	0.571	0.571	0.429	0.571	1.000	1.000	0.667	0.667	0.600	0.550	0.500	0.529	0.500	1.000
9	0.857	0.857	0.643	0.643	0.643	0.545	0.571	0.571	0.429	0.571	1.000	1.000	0.708	0.708	0.600	0.550	0.500	0.529	0.500	1.000
10	0.857	0.857	0.714	0.714	0.643	0.591	0.571	0.571	0.429	0.571	1.000	1.000	0.750	0.750	0.633	0.550	0.500	0.529	0.500	1.000
11	0.857	0.857	0.786	0.786	0.643	0.591	0.643	0.571	0.429	0.571	1.000	1.000	0.792	0.792	0.667	0.575	0.500	0.529	0.500	1.000
12	0.857	0.857	0.857	0.857	0.643	0.591	0.643	0.643	0.429	0.571	1.000	1.000	0.833	0.833	0.700	0.600	0.500	0.559	0.500	1.000
13	0.857	0.857	0.786	0.786	0.643	0.591	0.643	0.643	0.357	0.571	1.000	1.000	0.792	0.792	0.667	0.575	0.458	0.529	0.458	1.000
14	0.857	0.857	0.857	0.857	0.643	0.591	0.643	0.643	0.357	0.643	1.000	1.000	0.833	0.833	0.700	0.600	0.500	0.559	0.500	1.000
15	0.857	0.857	0.857	0.857	0.714	0.636	0.643	0.643	0.357	0.643	1.000	1.000	0.833	0.833	0.733	0.600	0.500	0.559	0.500	1.000
16	0.857	0.857	0.857	0.857	0.786	0.636	0.714	0.643	0.357	0.643	1.000	1.000	0.833	0.833	0.767	0.625	0.500	0.559	0.500	1.000
17	0.857	0.857	0.857	0.857	0.857	0.636	0.714	0.714	0.357	0.643	1.000	1.000	0.833	0.833	0.800	0.650	0.500	0.588	0.500	1.000
18	0.857	0.857	0.857	0.857	0.786	0.636	0.714	0.714	0.286	0.643	1.000	1.000	0.833	0.833	0.767	0.625	0.458	0.559	0.458	1.000
19	0.857	0.857	0.857	0.857	0.857	0.636	0.714	0.714	0.286	0.714	1.000	1.000	0.833	0.833	0.800	0.650	0.500	0.588	0.500	1.000
20	0.857	0.857	0.857	0.857	0.857	0.682	0.786	0.714	0.286	0.714	1.000	1.000	0.833	0.833	0.800	0.675	0.500	0.588	0.500	1.000
21	0.857	0.857	0.857	0.857	0.857	0.727	0.786	0.786	0.286	0.714	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.618	0.500	1.000
22	0.857	0.857	0.857	0.857	0.857	0.682	0.786	0.786	0.214	0.714	1.000	1.000	0.833	0.833	0.800	0.675	0.458	0.588	0.458	1.000
23	0.857	0.857	0.857	0.857	0.857	0.727	0.786	0.786	0.214	0.786	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.618	0.500	1.000
24	0.857	0.857	0.857	0.857	0.857	0.727	0.857	0.857	0.214	0.786	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.647	0.500	1.000
25	0.857	0.857	0.857	0.857	0.857	0.682	0.786	0.857	0.143	0.786	1.000	1.000	0.833	0.833	0.800	0.675	0.458	0.618	0.458	1.000
26	0.857	0.857	0.857	0.857	0.857	0.727	0.857	0.857	0.143	0.857	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.647	0.500	1.000
27	0.857	0.857	0.857	0.857	0.857	0.682	0.857	0.786	0.071	0.857	1.000	1.000	0.833	0.833	0.800	0.675	0.458	0.618	0.458	1.000
28	0.857	0.857	0.857	0.857	0.857	0.727	0.857	0.857	0.071	0.929	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.647	0.500	1.000
29	0.857	0.857	0.857	0.857	0.857	0.727	0.857	0.857	**0.000**	0.857	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.647	0.500	1.000

5.7. FACTORS AFFECTING THE SPFDI ALGORITHM

There are different factors which may affect the efficiency of the algorithm described in section 5.4, as well as the correctness of the results that are obtained from it. The following section delves into some of these factors.

5.7.1. Effect of Network Structure

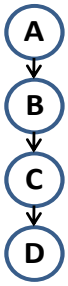
As seen in Chapter 4 (Section 4.2), the nodes in a Bayesian network can be linked in number of ways based on different criteria. The resultant network topology has a strong influence on the composition of the instantiation table. This, in turn can have a profound effect on the effectiveness of the SPFDI algorithm, in terms of how reliably it can identify and distinguish among faults.

A false alarm in the present context is defined as a situation where a particular sensor or process may be inadvertently flagged as being faulty by virtue of its column in the instantiation table constituting a subset of other columns corresponding to one or more of the remaining sensors/processes where real faults have occurred. Different types of network structures may result in different instances of false alarms. Intuitively, it may be expected that as the number of subsets formed for different supersets of columns (individually or in combinations) increases, the number of false alarms that may be expected by using such an instantiation table will also be higher[24].

Consider a simple system consisting of four variables *A*, *B*, *C*, and *D*. Figure 5-19 shows the different network structures that may arise, relating the nodes corresponding to these variables. Some elementary cases like a serial chain, diverging links, converging links, an acyclic loop, addition of redundant nodes, (Figure 5-19 (a)-(e) respectively) etc. have been considered for illustrative purposes; many other alternative structures are also possible.

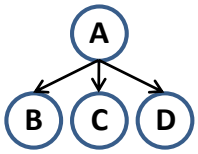
[24] As the number of faulty elements (sensors/processes) increase, it would be logical to expect that the instances of false alarms would increase (due to a greater number of unreliable sources of information as compared to reliable ones) except for the degenerate case when all the sensors/processes represented in the network are faulty.

(a)



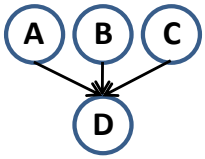
	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
1	*NI*	*NoI*			*PoI*		
2	*NI*		*NoI*		*PoI*	*PoI*	
3	*NI*			*NoI*	*PoI*	*PoI*	*PoI*
4		*NI*	*NoI*			*PoI*	
5		*NI*		*NoI*		*PoI*	*PoI*
6			*NI*	*NoI*			*PoI*

(b)



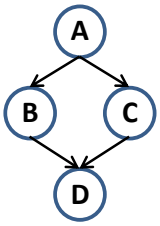
	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**A->C**	**A->D**
1	*NI*	*NoI*			*PoI*		
2	*NI*		*NoI*			*PoI*	
3	*NI*			*NoI*			*PoI*

(c)



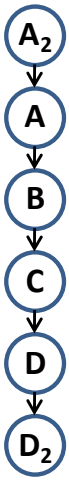
	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->D**	**B->D**	**C->D**
1	*NI*			*NoI*	*PoI*		
2		*NI*		*NoI*		*PoI*	
3			*NI*	*NoI*			*PoI*

(d)



	Sensors				Processes			
Step	**A**	**B**	**C**	**D**	**A->B**	**A->C**	**B->D**	**C->D**
1	*NI*	*NoI*			*PoI*			
2	*NI*		*NoI*			*PoI*		
3	*NI*			*NoI*	*PoI*	*PoI*	*PoI*	*PoI*
4		*NI*	*NI*	*NoI*			*PoI*	*PoI*

(e)



	Sensors						Processes				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
1	*NI*	*NoI*					*PoI*				
2	*NI*		*NoI*				*PoI*	*PoI*			
3	*NI*			*NoI*			*PoI*	*PoI*	*PoI*		
4	*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*	
5	*NI*					*NoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*
6		*NI*	*NoI*					*PoI*			
7		*NI*		*NoI*				*PoI*	*PoI*		
8		*NI*			*NoI*			*PoI*	*PoI*	*PoI*	
9		*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*
10			*NI*	*NoI*					*PoI*		
11			*NI*		*NoI*				*PoI*	*PoI*	
12			*NI*			*NoI*			*PoI*	*PoI*	*PoI*
13				*NI*	*NoI*					*PoI*	
14				*NI*		*NoI*				*PoI*	*PoI*
15					*NI*	*NoI*					*PoI*

Figure 5-19: Different Network Structures

The instantiation tables for these different configurations are also shown. The objective is to determine the extent to which the network structure and the resultant instantiation table cause the subset problem to occur (Section 5.5.3).

In Figure 5-19 (a), simply by observation, it can be seen that $Col_{A->B} \subseteq Col_A$[25], $Col_D \subseteq (Col_C \cup Col_{B->C})$, and $Col_{C->D} \subseteq (Col_C \cup Col_{B->C})$. This implies that with either a fault in the sensor corresponding to *A*, or with a fault in the process *A->B*, the algorithm results indicate a potential fault in both the sensor and the process. Similarly, with a fault in the sensor corresponding to *C* and a simultaneous fault in the *B->C* process, in addition to flagging a possible fault in both these elements, the algorithm also wrongly signals a potential fault in the sensor corresponding to *D* and the process *C->D*.

In Figure 5-19 (b), it is observed that $Col_B \subseteq Col_A$, $Col_C \subseteq Col_A$, $Col_D \subseteq Col_A$, $Col_{A->B} \subseteq Col_A$ and Col_B, $Col_{A->C} \subseteq Col_A$ and Col_C, and $Col_{A->D} \subseteq Col_A$ and Col_D. In this case, due to the numerous subsets, the potential for false alarms is quite high and without the addition of redundant nodes (representing additional sensors) or altering the network structure, the SPFDI algorithm cannot reliably distinguish between the different types of faults. Similarly, for Figure 5-19 (c), (d), it can be seen that the formation of subsets can hamper the algorithm in clearly distinguishing between sensor and process faults with any unanticipated deviations in sensor readings.

The occurrence of false alarms may be reduced via careful construction of the instantiation table by the domain expert. This may be achieved, for instance, by introducing different types of instantiations (for instance, utilizing both predictive/ diagnostic reasoning), choosing the order in which the nodes are instantiated, performing an exhaustive examination of all possible combinations of columns and their subsets, etc. The goal is to modify the instantiation table structure to prevent subsets from being formed as far as possible. However, despite all these efforts, situations may still exist where it may not be possible to eliminate the potential for false alarms completely. In such cases, it would be desirable to calculate the risk associated with using the available

[25] In this case, since the two columns are identical i.e. $Col_{A->B} \equiv Col_A$, the relation that $Col_A \subseteq Col_{A->B}$ can also be considered to be equally valid

instantiation table structure with the SPFDI algorithm[26]. This is particularly vital if there is a possibility that a key sensor or a process may end up being flagged as often faulty, despite no real faults occurring. Armed with this information, the domain expert may be able to alter the instantiation table before implementing the SPFDI algorithm, in order to ensure its maximum effectiveness and accuracy in detecting and isolating faults. The following section describes a possible approach to estimate beforehand, the risk of false alarms occurring based on the composition of the instantiation table.

Consider the network shown in Figure 5-19 (e). This network consists of six nodes or six sensors in an actual system. The network also has five different links representing the processes in the system. Thus, there are a total of eleven possible single sensor or process faults that may occur in the system represented by this network. With the addition of nodes A_2 and D_2 representing redundant sensors for the variables corresponding to the original root and leaf nodes A and D (Figure 5-19 (a)), the subset issue is alleviated to a large extent, especially for single faults in either the sensors or the processes since all the columns in the instantiation table are made distinct[27]. Excluding the ideal processes A_2->A and D->D_2, there are nine possible elements in which real faults may occur.

Each of these elements may be considered potentially faulty or not. This results in a total of $\sum_{r=0}^{9}({}^{9}C_r)$ combinations of faults, where r represents the number of faulty sensors and/or processes i.e. 512 possible combinations of faults (2^9) ranging from no faults (i.e. $r = 0$ which occurs in only one combination) to the extreme situation when all the sensors and processes are faulty (r=9, which can again happen in only one combination).

---

[26] For instance, if results from the SPFDI algorithm are indicated to the operator via warning lamps on the system control panel, showing the existence of a fault, and if the light corresponding to a particular sensor/process flashes, the operator needs to be informed of the odds that the warning light is due to an actual fault in the sensor/process rather than a false alarm caused by the subset issue.

[27] As explained in Section 5.5.4, even though the columns corresponding to A_2 /A_2->A are identical and the columns corresponding to D_2 /D->D_2, the processes A_2->A and D->D_2 represent ideal processes which can never be faulty; their belief values are always 1 and are never modified during the algorithm execution.

Table 5-29: All Possible Combinations for One, Two and Three Faulty Elements

One Fault	
1	*C->D*
2	*B->C*
3	***A->B***
4	D_2
5	*D*
6	*C*
7	*B*
8	*A*
9	A_2

Two Faults						
1	*B->C*	*C->D*		**19**	*B*	D_2
2	*A->B*	*C->D*		**20**	*B*	*D*
3	*A->B*	*B->C*		**21**	*B*	*C*
4	D_2	*C->D*		**22**	*A*	*C->D*
5	D_2	*B->C*		**23**	*A*	*B->C*
6	D_2	*A->B*		**24**	*A*	*A->B*
7	*D*	*C->D*		**25**	*A*	D_2
8	*D*	*B->C*		**26**	*A*	*D*
9	***D***	***A->B***		**27**	*A*	*C*
10	*D*	D_2		**28**	*A*	*B*
11	*C*	*C->D*		**29**	A_2	*C->D*
12	*C*	*B->C*		**30**	A_2	*B->C*
13	*C*	*A->B*		**31**	A_2	*A->B*
14	*C*	D_2		**32**	A_2	D_2
15	*C*	*D*		**33**	A_2	*D*
16	*B*	*C->D*		**34**	A_2	*C*
17	*B*	*B->C*		**35**	A_2	*B*
18	*B*	*A->B*		**36**	A_2	*A*

Three Faults																			
1	*A->B*	*B->C*	*C->D*		**22**	*B*	*A->B*	*C->D*		**43**	*A*	*D*	*B->C*		**64**	A_2	*D*	*B->C*
2	D_2	*B->C*	*C->D*		**23**	*B*	*A->B*	*B->C*		**44**	*A*	*D*	*A->B*		**65**	A_2	*D*	*A->B*
3	D_2	*A->B*	*C->D*		**24**	*B*	D_2	*C->D*		**45**	*A*	*D*	D_2		**66**	A_2	*D*	D_2
4	D_2	*A->B*	*B->C*		**25**	*B*	D_2	*B->C*		**46**	*A*	*C*	*C->D*		**67**	A_2	*C*	*C->D*
5	*D*	*B->C*	*C->D*		**26**	*B*	D_2	*A->B*		**47**	*A*	*C*	*B->C*		**68**	A_2	*C*	*B->C*
6	*D*	*A->B*	*C->D*		**27**	*B*	*D*	*C->D*		**48**	*A*	*C*	*A->B*		**69**	A_2	*C*	*A->B*
7	*D*	*A->B*	*B->C*		**28**	*B*	*D*	*B->C*		**49**	*A*	*C*	D_2		**70**	A_2	*C*	D_2
8	*D*	D_2	*C->D*		**29**	***B***	***D***	***A->B***		**50**	*A*	*C*	*D*		**71**	A_2	*C*	*D*
9	*D*	D_2	*B->C*		**30**	*B*	*D*	D_2		**51**	*A*	*B*	*C->D*		**72**	A_2	*B*	*C->D*
10	*D*	D_2	*A->B*		**31**	*B*	*C*	*C->D*		**52**	*A*	*B*	*B->C*		**73**	A_2	*B*	*B->C*
11	*C*	*B->C*	*C->D*		**32**	*B*	*C*	*B->C*		**53**	*A*	*B*	*A->B*		**74**	A_2	*B*	*A->B*
12	*C*	*A->B*	*C->D*		**33**	*B*	*C*	*A->B*		**54**	*A*	*B*	D_2		**75**	A_2	*B*	D_2
13	*C*	*A->B*	*B->C*		**34**	*B*	*C*	D_2		**55**	*A*	*B*	*D*		**76**	A_2	*B*	*D*
14	*C*	D_2	*C->D*		**35**	*B*	*C*	*D*		**56**	*A*	*B*	*C*		**77**	A_2	*B*	*C*
15	*C*	D_2	*B->C*		**36**	*A*	*B->C*	*C->D*		**57**	A_2	*B->C*	*C->D*		**78**	A_2	*A*	*C->D*
16	*C*	D_2	*A->B*		**37**	*A*	*A->B*	*C->D*		**58**	A_2	*A->B*	*C->D*		**79**	A_2	*A*	*B->C*
17	*C*	*D*	*C->D*		**38**	*A*	*A->B*	*B->C*		**59**	A_2	*A->B*	*B->C*		**80**	A_2	*A*	*A->B*
18	*C*	*D*	*B->C*		**39**	*A*	D_2	*C->D*		**60**	A_2	D_2	*C->D*		**81**	A_2	*A*	D_2
19	*C*	*D*	*A->B*		**40**	*A*	D_2	*B->C*		**61**	A_2	D_2	*B->C*		**82**	A_2	*A*	*D*
20	*C*	*D*	D_2		**41**	*A*	D_2	*A->B*		**62**	A_2	D_2	*A->B*		**83**	A_2	*A*	*C*
21		*B->C*	*C->D*		**42**	*A*	*D*	*C->D*		**63**	A_2	*D*	*C->D*		**84**	A_2	*A*	*B*

Table 5-29 shows some such possible combinations in which one (9), two (36) or three (84) faults may occur (similar combinations may be determined for a higher number of faults). Due to the completely distinct columns of the instantiation table in Figure 5-19 (e), no subsets are formed for any of the columns when they are considered individually. Thus, if a single sensor or a single process fault occurs, the SPFDI algorithm, when applied to this network structure with the aforesaid instantiation table can always precisely identify the occurrence of those faults without false alarms being raised for any of the other sensors or processes. For instance, considering row 3 in the column corresponding to single faults in Table 5-29 when a real fault occurs in the *A->B* process, the SPFDI algorithm will be able to identify it uniquely, resulting in $W_{A->B} = 0$, without signaling any false alarms.

Now, consider row 9 in the column in Table 5-29 corresponding to the occurring of two faults in different elements of the network. This indicates the occurrence of a real fault in the sensor corresponding to *D* and the process *A->B* simultaneously. Similarly row 8 (combination no. 29) in the column in Table 5-29, corresponding to the occurring of three faults, indicates the occurrence of real faults in the sensors corresponding to *B* and *D* and the process *A->B* simultaneously. In both these cases, in order to determine the possibility of false alarms for other sensors or processes, it is necessary to determine whether the columns corresponding to those remaining sensors or processes form a subset of $Col_D \cup Col_{A->B}$ or $Col_B \cup Col_D \cup Col_{A->B}$ respectively. The algorithm shown below can help determine the subsets that would be formed for different combinations of faults.

Objective	Given a Bayesian network consisting of a set of nodes $X = \{X_1, X_2, .. X_n\} = \{X_i\}$, where $i = 1, 2, \ldots n$, n is the total number of nodes in the network
Inputs	• Number of sensors n, Number of processes m • Instantiation table based on the network structure with $(n+m)$ columns
Outcome	The different subsets formed by the columns in the instantiation table taken individually and in combinations

Algorithm

// *Function to determine whether a column (say, col_1) in an instantiation table is a subset of another column (say, col_2)*

function is_subset (*col_1*, *col_2*)

{

Loop through *r* the number of rows in the instantiation table

a. Determine if the {*r*, *col_1*}th cell in *col_1* and the {*r*, *col_2*}th cell in *col_2* are both occupied (by a *NI*, *NoI* or *PoI* corresponding to that column as the case may be)

b. If so, store the row number *r*

End loop *r*

If the stored row numbers are such that they are exactly the same as the row numbers corresponding to the occupied cells in *col_1*, then *col_1* can be said to be a subset of *col_2*

}

Main ()*:*

Loop through $k = 1, 2, \ldots n+m$ i.e. the number of columns in the instantiation table

Loop through $i = 1, 2, \ldots n,$ the number of columns corresponding to sensors/nodes in the network ($i \neq k$)

Determine if the k^{th} column is a subset of the i^{th} column using the function *is_subset* (*k*, *i*)

Store the result

End Loop *i*

Loop through $j = 1, 2, \ldots m$ the number of columns corresponding to processes/links in the network other than the ideal processes ($j \neq k$)

Determine if the k^{th} column is a subset of the j^{th} column using the function *is_subset* (*k*, *j*)

Store the result

End Loop j

Loop through $i = 1, 2, \ldots n;\ i \neq k$

Loop through $j = 1, 2, \ldots m;\ j \neq k$

a. Determine all possible combinations in which the i and j columns can be combined. The number of columns in these combinations range from 2 to $n+m-1$

b. Merge the columns in each combination into a temporary t^{th} column

c. Determine if the k^{th} column is a subset of the t^{th} column using the function

is_subset (k, t)

d. Store the result

End Loop i

End Loop j

End Loop k

Implementing the algorithm described above, Table 5-30 shows the different possible combinations of columns corresponding to real faults in different sensors/processes (supersets) and the subsets of columns that may be formed when these columns are combined. Certain combinations of faults shown in Table 5-29 may not cause any false alarms if a combination of the columns corresponding to those faulty sensors/processes does not have any subsets in the instantiation table. For instance, in the example considered earlier, with two simultaneous faults in the sensor corresponding to D and the process $A->B$, the combination of columns $Col_D \cup Col_{A->B}$ has no subsets in the instantiation table (Figure 5-19 (e)). Thus, this combination of faults does not generate any false alarms in any of the other elements. However, with three simultaneous faults in the sensors corresponding to B and D as well as the process $A->B$, it can be seen from Table 5-30 that the combination of the columns corresponding to these elements has one subset in the instantiation table i.e. $Col_{B->C} \subseteq \{Col_B \cup Col_D \cup Col_{A->B}\}$. Thus, with this combination of faults, the SPFDI algorithm will also falsely signal a potential fault in the process $B->C$.

Table 5-30: Combinations of Multiple (Two or Three) Faults for which Subsets are Formed in the Instantiation Table

F_i = Faults, i =2, 3…9

SS = Subset

Two Faults			
	$F_1 \cup F_2$		SS
1	B->C	C->D	C
2	A->B	B->C	B
3	D_2	C->D	D
4	D	C->D	D_2
5	D	D_2	C->D
6	C	C->D	B->C
7	C	B->C	C->D
8	B	B->C	A->B
9	B	A->B	B->C
10	A	A->B	A_2
11	A_2	A->B	A
12	A_2	A	A->B

Three Faults																							
	$F_1 \cup F_2 \cup F_3$			SS		$F_1 \cup F_2 \cup F_3$			SS		$F_1 \cup F_2 \cup F_3$			SS		$F_1 \cup F_2 \cup F_3$			SS
1	A->B	B->C	C->D	B	14	C	D	C->D	D_2	28	B	C	A->B	B->C	44	A_2	A->B	C->D	A
				C					B->C					C->D	45	A_2	D_2	C->D	D
2	D_2	B->C	C->D	C	15	C	D	B->C	D_2	29	A	B->C	C->D	C	46	A_2	D_2	A->B	A
				D					C->D	30	A	A->B	C->D	A_2	47	A_2	D	C->D	D_2
3	D_2	A->B	C->D	D	16	C	D	D_2	B->C	31	A	A->B	B->C	A_2	48	A_2	D	A->B	A
4	D_2	A->B	B->C	B					C->D					B	49	A_2	D	D_2	C->D
5	D	B->C	C->D	C	17	B	B->C	C->D	C	32	A	D_2	C->D	D	50	A_2	C	C->D	B->C
				D_2					A->B	33	A	D_2	A->B	A_2	51	A_2	C	B->C	C->D
6	D	A->B	C->D	D_2	18	B	A->B	C->D	C	34	A	D	C->D	D_2	52	A_2	C	A->B	A
7	D	A->B	B->C	B					B->C	35	A	D	A->B	A_2	53	A_2	B	B->C	A
8	D	D_2	B->C	C	19	B	D_2	C->D	D	36	A	D	D_2	C->D					A->B
				C->D	20	B	D_2	B->C	A->B	37	A	C	C->D	B->C	54	A_2	B	A->B	A
9	D	D_2	A->B	C->D	21	B	D_2	A->B	B->C	38	A	C	B->C	C->D					B->C
10	C	A->B	C->D	B	22	B	D	C->D	D_2	39	A	C	A->B	A_2	55	A_2	A	C->D	A->B
				B->C	23	B	D	B->C	A->B	40	A	B	B->C	A_2	56	A_2	A	B->C	B
11	C	A->B	B->C	B	**24**	***B***	***D***	***A->B***	***B->C***					A->B					A->B
				C->D	25	B	D	D_2	C->D	41	A	B	A->B	A_2	57	A_2	A	D_2	A->B
12	C	D_2	C->D	D	26	B	C	C->D	A->B					B->C	58	A_2	A	D	A->B
				B->C					B->C	42	A_2	B->C	C->D	C	59	A_2	A	C	A->B
13	C	D_2	B->C	D	27	B	C	B->C	A->B	43	A_2	A->B	B->C	A	60	A_2	A	B	A->B
				C->D					C->D					B					B->C

For the network shown in Figure 5-19 (e), from Table 5-30, it can be seen that the number of supersets formed when columns corresponding to two faults (sensors and/or processes) are combined is 12. Similarly the number of supersets formed for combinations of three faults is 60. The number of such supersets formed can be used to define the effectiveness of the SPFDI algorithm, by estimating probability of false alarms that occur as a fraction of all possible combinations of faults.

Two types of errors that may occur during the fault detection and isolation process are: a non-faulty sensor/process may be classified as faulty, or a faulty sensor/process may remain undetected. These are referred to as type I and type II errors respectively in the literature [Cohen, 1995]. The definitions are based on statistical hypothesis testing using the null hypothesis. Formally, type I error (false positive) refers to the rejection of the null hypothesis when it is true and type II error (false negative) refers to the acceptance of the null hypothesis when it is actually false. For the SPFDI procedure, the null hypothesis is that a particular sensor or a process is operating correctly. The discussion so far, thus pertains to the calculation of type I errors[28].

Table 5-31: Probability of False Alarms

No. of Faults	Total Combinations	Supersets (combinations of real faults that form subsets)	*P* (False Alarms)
0	1	0	0.00
1	9	0	0.00
2	36	12	0.33
3	84	60	0.71
4	126	110	0.87
5	126	119	0.94
6	84	79	0.94
7	36	34	0.94
8	9	9	1.00
9	1	0	0.00

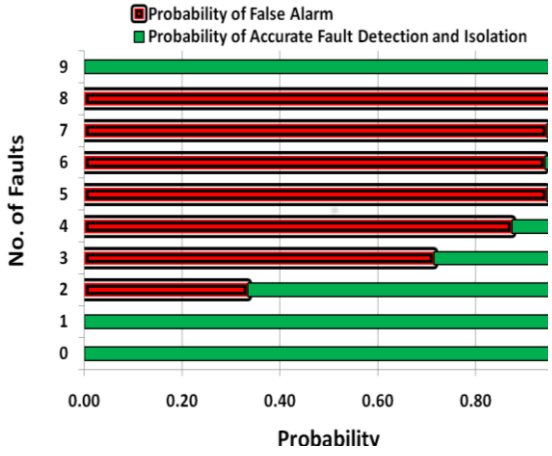


Figure 5-20: Probability of False Alarms

[28] The calculation of type II errors requires more extensive test data and will be addressed in future research.

Table 5-31 and Figure 5-20 show the probability of false alarms that may occur with different number of faults in the elements of the network shown in Figure 5-19 (e). Thus, it can be seen that the SPFDI algorithm is very effective when there are single sensor or process faults (excluding the cases where all or none of the sensors and processes are faulty). In scenarios with multiple sensor and/or process faults, it is clear that the probability of false alarms increases as the number of faulty elements increases. In such cases, the probability of false alarms for individual sensors or processes may be calculated as described below.

Since a particular sensor or a process may actually be faulty or it may be wrongly classified as being faulty due to the subset issue, all such cases must be taken into account while estimating the probability of false alarms for any given sensor or process. In order to do so, the columns in Table 5-30 may be re-arranged by grouping together the common elements in a given combination of sensor/process faults as shown in Table 5-32 and Table 5-33. Consider rows 1, 2 and 3 in Table 5-32 representing the combinations of two faults and the corresponding subset. Each time a fault occurs in any two of the three elements i.e. the two processes *B->C*, *C->D* or the sensor corresponding to *C*, the SPFDI algorithm will also generate a false alarm for the third element due to the formation of a subset. Hence, the probability for each of these elements being actually faulty and the probability of a false alarm can be calculated as follows:

$$P(Z)_{AF} = \frac{n_{Z_{AF}}}{n_{Z_{AF}} + n_{Z_{FA}}} \qquad \text{Eq. (5-2)}$$

$$P(Z)_{FA} = 1 - P(Z)_{AF}$$

where *Z* represents a particular sensor or process, $n_{Z_{AF}}$ is the number of times the element *Z* occurs in Table 5-32 due to an actual fault, and $n_{Z_{FA}}$ is the number of times Z occurs in Table 5-32 due to a false alarm for *Z*, $P(Z)_{AF}$ is the probability that *Z* is actually faulty and $P(Z)_{FA}$ is the probability of a false alarm being raised for *Z*. The probability of actual faults and false alarms when two simultaneous faults occur are as shown in Table 5-34 and Table 5-35 respectively.

Table 5-32 : Combinations of Two Faults

	Actual Faults		Subset
1	*B->C*	*C->D*	*C*
2	*C*	*C->D*	*B->C*
3	*C*	*B->C*	*C->D*
4	*A->B*	*B->C*	*B*
5	*B*	*B->C*	*A->B*
6	*B*	*A->B*	*B->C*
7	D_2	*C->D*	*D*
8	*D*	*C->D*	D_2
9	*D*	D_2	*C->D*
10	*A*	*A->B*	A_2
11	A_2	*A->B*	*A*
12	A_2	*A*	*A->B*

Table 5-33 : Combinations of Three Faults

	Actual Faults			Subset
1	*A->B*	*B->C*	*C->D*	*B* *C*
2	*B*	*C*	*B->C*	*A->B* *C->D*
3	*B*	*C*	*C->D*	*A->B* *B->C*
4	*B*	*C*	*A->B*	*B->C* *C->D*
5	*B*	*B->C*	*C->D*	*C* *A->B*
6	*B*	*A->B*	*C->D*	*C* *B->C*
7	*C*	*A->B*	*C->D*	*B* *B->C*
8	*C*	*A->B*	*B->C*	*B* *C->D*
9	*D2*	*A->B*	*C->D*	*D*
10	*D*	*A->B*	*C->D*	*D2*
11	*D*	*D2*	*A->B*	*C->D*
12	*B*	*D2*	*C->D*	*D*
13	*B*	*D*	*C->D*	*D2*
14	*B*	*D*	*D2*	*C->D*
15	*B*	*D2*	*B->C*	*A->B*
16	*B*	*D2*	*A->B*	*B->C*
17	*D2*	*A->B*	*B->C*	*B*
18	*B*	*D*	*B->C*	*A->B*
19	*B*	*D*	*A->B*	*B->C*
20	*D*	*A->B*	*B->C*	*B*
21	A_2	*A->B*	*B->C*	*A* *B*
22	A_2	*A*	*B->C*	*B* *A->B*
23	A_2	*A*	*B*	*A->B* *B->C*
24	A_2	*B*	*B->C*	*A* *A->B*
25	A_2	*B*	*A->B*	*A* *B->C*
26	*A*	*A->B*	*B->C*	A_2 *B*
27	*A*	*B*	*B->C*	A_2 *A->B*
28	*A*	*B*	*A->B*	A_2 *B->C*
29	*A*	*B->C*	*C->D*	*C*
30	*A*	*C*	*C->D*	*B->C*
31	*A*	*C*	*B->C*	*C->D*
32	*A*	D_2	*C->D*	*D*
33	*A*	*D*	*C->D*	D_2
34	*A*	*D*	D_2	*C->D*
35	*A*	D_2	*A->B*	A_2
36	A_2	D_2	*A->B*	*A*
37	*A2*	*A*	D_2	*A->B*
38	*A*	*D*	*A->B*	A_2
39	A_2	*D*	*A->B*	*A*
40	A_2	*A*	*D*	*A->B*
41	*C*	D_2	*C->D*	*D* *B->C*
42	*C*	D_2	*B->C*	*D* *C->D*
43	*C*	*D*	*C->D*	D_2 *B->C*
44	*C*	*D*	*B->C*	D_2 *C->D*
45	*C*	*D*	D_2	*B->C* *C->D*
46	D_2	*B->C*	*C->D*	*C* *D*
47	*D*	*B->C*	*C->D*	*C* D_2
48	*D*	D_2	*B->C*	*C* *C->D*
49	*A*	*C*	*A->B*	A_2
50	A_2	*A*	*C*	*A->B*
51	A_2	*C*	*A->B*	*A*
52	A_2	*B->C*	*C->D*	*C*
53	A_2	*C*	*C->D*	*B->C*
54	A_2	*C*	*B->C*	*C->D*
55	A_2	*A->B*	*C->D*	*A*
56	*A*	*A->B*	*C->D*	A_2
57	A_2	*A*	*C->D*	*A->B*
58	A_2	D_2	*C->D*	*D*
59	A_2	*D*	*C->D*	D_2
60	A_2	*D*	D_2	*C->D*

Considering the first row in Table 5-34 and Table 5-35, the values may be interpreted as follows (Only the shaded cells denote elements which are faulty): If a simultaneous fault occurs in any two of the three elements- the sensor corresponding to node *C*, and the

processes *B->C* and *C->D*, then the probability that there is an actual fault in any of them is 0.6667 and the probability that the indicated fault is a false alarm is 0.3333 (note that these values are valid only for the specified combination of faults). Similar values can be calculated for three or more faults. For three simultaneous faults, Table 5-36 shows the probability of occurrence of actual faults and Table 5-37 shows the probability of false alarms.

Table 5-34: Probability of Actual Faults for Combinations of Two Faulty Elements

$P(A_2)_{AF}$	$P(A)_{AF}$	$P(B)_{AF}$	$P(C)_{AF}$	$P(D)_{AF}$	$P(D_2)_{AF}$	$P(A->B)_{AF}$	$P(B->C)_{AF}$	$P(C->D)_{AF}$
			0.6667				0.6667	0.6667
		0.6667				0.6667	0.6667	
				0.6667	0.6667			0.6667
0.6667	0.6667					0.6667		

Table 5-35: Probability of False Alarms Combinations of Two Faulty Elements

$P(A_2)_{FA}$	$P(A)_{FA}$	$P(B)_{FA}$	$P(C)_{FA}$	$P(D)_{FA}$	$P(D_2)_{FA}$	$P(A->B)_{FA}$	$P(B->C)_{FA}$	$P(C->D)_{FA}$
			0.3333				0.3333	0.3333
		0.3333				0.3333	0.3333	
				0.3333	0.3333			0.3333
0.3333	0.3333					0.3333		

Table 5-36: Probability of Actual Faults for Combinations of Three Faulty Elements

$P(A_2)_{AF}$	$P(A)_{AF}$	$P(B)_{AF}$	$P(C)_{AF}$	$P(D)_{AF}$	$P(D_2)_{AF}$	$P(A->B)_{AF}$	$P(B->C)_{AF}$	$P(C->D)_{AF}$
		0.625	0.625			0.625	0.5	0.625
0.625	0.625	0.625				0.5	0.625	
			0.625	0.625	0.625		0.625	0.5
				0.6667	0.6667	1.000	1.000	
		1.000		0.6667	0.6667			0.6667
		0.6667			1.0000	0.6667	0.6667	
		0.6667		1.0000		0.6667	0.6667	
	1.0000		0.6667				0.6667	0.6667
	1.0000			0.6667	0.6667			0.6667
0.6667	0.6667				1.000	0.6667		
0.6667	0.6667			1.0000		0.6667		
0.6667	0.6667		1.000			0.6667		
1.0000			0.6667				0.6667	0.6667
0.6667	0.6667					0.6667		1.000
1.000				0.6667	0.6667			0.6667

Table 5-37: Probability of False Alarms Combinations of Three Faulty Elements

$P(A_2)_{FA}$	$P(A)_{FA}$	$P(B)_{FA}$	$P(C)_{FA}$	$P(D)_{FA}$	$P(D_2)_{FA}$	$P(A->B)_{FA}$	$P(B->C)_{FA}$	$P(C->D)_{FA}$
		0.375	0.375			0.375	0.5	0.375
0.375	0.375	0.375				0.5	0.375	
			0.375	0.375	0.375		0.375	0.5
				0.3333	0.3333	0	0	
		0		0.3333	0.3333			0.3333
		0.3333			0	0.3333	0.3333	
		0.3333		0		0.3333	0.3333	
	0		0.3333				0.3333	0.3333
	0			0.3333	0.3333			0.3333
0.3333	0.3333				0	0.3333		
0.3333	0.3333			0		0.3333		
0.3333	0.3333		0			0.3333		
0			0.3333				0.3333	0.3333
0.3333	0.3333					0.3333		0
0				0.3333	0.3333			0.3333

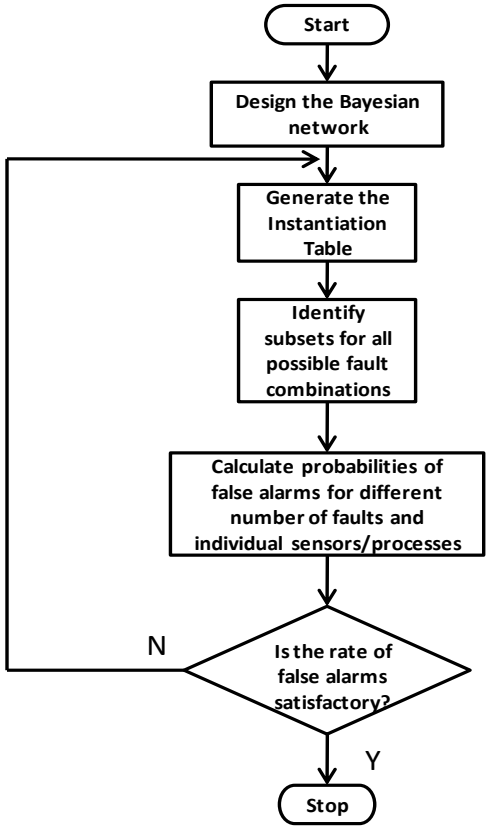


Figure 5-21: Reducing false alarms

As the values for probability of false alarms can be generated beforehand, if a particular combination of faults occurs during operation, then based on the output of the SPFDI algorithm and such charts, the system operator can make an informed decision about the fault status of a particular set of sensors/processes. Once a network structure has been designed for the system, and its associated instantiation table has been created, the values shown in Table 5-31, Figure 5-20 and Table 5-34 to Table 5-37, may be generated. If it is found that the proportion of false alarms is unusually high, then the domain expert may revise the composition of the

instantiation table to reduce the chances of false alarms (Figure 5-21).

5.7.2. Reducing Instantiations

Although the SPFDI algorithm described thus far can be considered an anytime algorithm in that it gives an indication of the health of the different sensors and processes in the system at any given instant, as seen in Sections 5.5 and 5.6, a number of instantiations as well as inferences have to be done even to complete a single validation cycle that essentially deals with only one sample of data. To declare a particular sensor or a process to be faulty with some certainty requires the analysis of multiple samples over a given period of time, which may require considerable computation time and resources. This issue may be greatly alleviated if the time taken for each individual validation cycle can be reduced somehow.

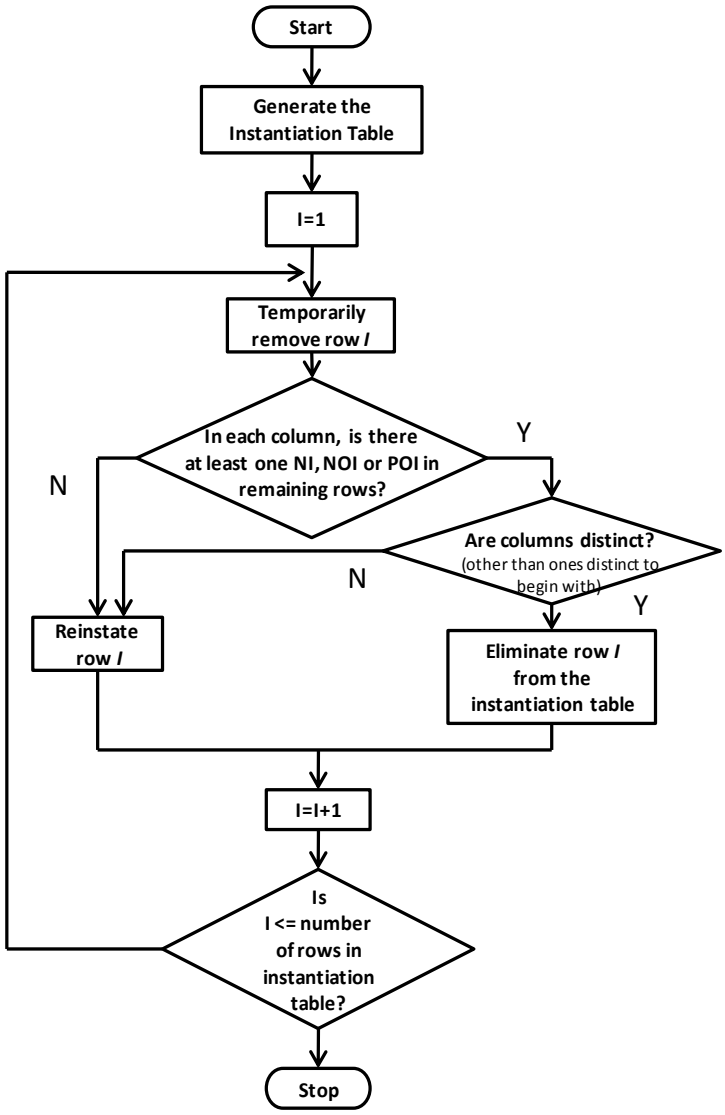


Figure 5-22: Reducing the Size of the Instantiation Table

A potential approach to this is to reduce the number of instantiations in the instantiation table. The following section describes one possible approach that can be followed. Consider an instantiation table with *n* rows (Table 5-38). Starting with the first row and working down towards the end of the instantiation table (represented by the counter *I =1, 2…n* in Figure 5-22), a row may be eliminated only after ensuring that both the following conditions are satisfied:

a) There are no identical columns and the columns do not form subsets of other columns

b) In each column of the instantiation table, there is at least one row that includes the sensor or the process represented by that column as a *NI*, *NoI* or *PoI* as the case may be.

If either of the above conditions is not satisfied then that row cannot be eliminated from the instantiation table, since it may potentially lead to false alarms from the fault detection and isolation algorithm.

The first condition addresses the issue of being able to uniquely identify potential faults in various sensors and processes. The second condition ensures that even after removing certain rows in the instantiation table, the algorithm still retains the ability to identify every faulty sensor or process that is a part of the instantiation table. Table 5-39 shows the result of eliminating some rows from the original instantiation table (more rows could be potentially eliminated provided the two conditions mentioned earlier are always satisfied).

Although having a smaller instantiation table may lead to a faster fault detection and isolation cycle, there may be potential pitfalls in reducing the number of instantiations as described above. Since the values of ε_S and ε_P, the incremental values by which the W_S and W_P belief values in the condition of the sensors and processes are modified, depend on the number of rows a particular sensor or process occurs in the instantiation table i.e. n_S or n_p, the granularity of change in the belief values is reduced by reducing the number of instantiations (since the values of n_S or n_p decrease, the belief values change by larger amounts as given by Eq.(5-1)).

Depending on the threshold values (or bin sizes corresponding to the different states of a node) that are used to determine whether an observed deviation in the sensor

reading is acceptable or not, or whether it is the result of a sensor or a process fault, such a change might result in more instances of false alarms, even after multiple samples have been analyzed[29]. The tradeoff, in terms of speed of execution and the potential for false alarms, between using a full instantiation table and a reduced instantiation table needs to be investigated further in the future.

Table 5-38: Original Instantiation Table

	Sensors						Processes				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
1	*NI*	*NoI*					*PoI*				
2	*NI*		*NoI*				*PoI*	*PoI*			
3	*NI*			*NoI*			*PoI*	*PoI*	*PoI*		
4	*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*	
5	*NI*					*NoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*
6		*NI*	*NoI*					*PoI*			
7		*NI*		*NoI*				*PoI*	*PoI*		
8		*NI*			*NoI*			*PoI*	*PoI*	*PoI*	
9		*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*
10			*NI*	*NoI*					*PoI*		
11			*NI*		*NoI*				*PoI*	*PoI*	
12			*NI*			*NoI*			*PoI*	*PoI*	*PoI*
13				*NI*	*NoI*					*PoI*	
14				*NI*		*NoI*				*PoI*	*PoI*
15					*NI*	*NoI*					*PoI*

Table 5-39: Example of a Partially Reduced Instantiation Table

	Sensors						Processes				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
1	*NI*	*NoI*					*PoI*				
6		*NI*	*NoI*					*PoI*			
7		*NI*		*NoI*				*PoI*	*PoI*		
8		*NI*			*NoI*			*PoI*	*PoI*	*PoI*	
9		*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*
10			*NI*	*NoI*					*PoI*		
11			*NI*		*NoI*				*PoI*	*PoI*	
12			*NI*			*NoI*			*PoI*	*PoI*	*PoI*
13				*NI*	*NoI*					*PoI*	
14				*NI*		*NoI*				*PoI*	*PoI*
15					*NI*	*NoI*					*PoI*

[29] If the thresholds are too restrictive or the bin sizes are too small, then even with a small deviation in the sensor reading outside of these limits, the corresponding belief value would be penalized heavily

5.7.3. Effect of Sensor Characteristics

The operational characteristics like accuracy, resolution, etc. of all the sensors in the system can play an important role in determining the overall efficacy of the SPFDI algorithm. The terms resolution and accuracy are often used interchangeably. However, the two quantities are quite distinct. The resolution of a sensor is the smallest incremental change in the physical input that can be reliably detected by the sensing element to produce a detectable change at the output [Pallas-Areny and Webster, 2001]. Resolution is dictated mainly by the sensor physics i.e. a sensor cannot discern any input signals that occur between its smallest subdivisions to produce a corresponding output. It is a fixed value typically specified in absolute terms in the same units as the measurand or in terms of bits (especially for sensors with digital output).

For instance, consider an incremental encoder that produces a digital output signal of 1024 counts per revolution. The resolution for this sensor may be specified as 10 bits (2^{10}) or 360/1024= 0.3515^{o}. In some cases, to enable interfacing with data acquisition equipment, the analog voltage or current output from a sensor is converted to digital form using an A/D converter which may dictate the resolution. For example, a potentiometer (as a voltage divider) to measure voltage theoretically has an infinite resolution since any motion of the slider will change the output voltage. However, if it is used in conjunction with a 10 bit resolution A/D converter in the 0-10V range, the resolution would be in steps of 9.76 mV[30].

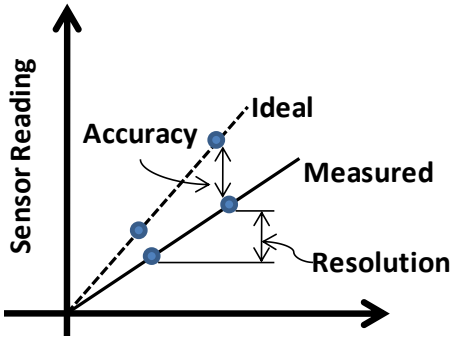


Figure 5-23: Sensor Resolution and Accuracy

[30] The limit to how exactly the output voltage can be measured is also determined by the limit to which the resolution can be discretized. Over the 0-10V range, a measurement made using such an arrangement can be no more accurate than 1 LSB or 9.76 mV (or ±1/2 LSB i.e. ±4.88 mV)

The accuracy of a sensor is defined as the maximum difference between the 'true' or ideal value of the measurand and the actual/measured sensor output [Fraden, 2004][31]. Accuracy can be specified in numerous ways depending on the type of sensor- in bits, as a % of the actual sensor output, as a % of the full scale range of the sensor, or in absolute terms in measurement units (for simplicity, only the sensor accuracy is considered in the present discussion[32]). It must be noted that sensor accuracy is just one part of the overall accuracy of the complete data acquisition system. Additional errors may be introduced by other components in the data acquisition system).

In general, accuracy and resolution are not directly related. An inaccurate sensor may have a high resolution, and vice versa. The accuracy of a sensor is determined through calibration against a known standard or against a gold-standard reference sensor. Hence, although the sensor accuracy is not directly related to the sensor resolution, the accuracy level can never be better than the resolution of the sensor (because the sensor can measure a value only to the limit of granularity imposed by its resolution[33]). Typically, the resolution of a sensor, by design, is many times better than the accuracy (sometimes an order of magnitude).

The quality of sensors corresponding to the *NI* determines which state of these nodes is set as evidence. This, in turn dictates the value of the *NoI* inferred by inferencing (the discretization as well as the conditional probability distributions of the intermediate nodes between the *NI* and the *NoI* are other factors that also influence the final inferred value). Similarly the quality of sensors corresponding to the *NoI* influences the results of comparison of the inferred value and the indicated value (when a decision is made if the observed deviation in the sensor value is within acceptable limits or not, in order to alter

[31] In the present discussion, accuracy is taken to encompass a region around the sensor reading with 95% confidence value.

[32] The accuracy of a sensor may also be influenced by other characteristics like hysteresis, linearity, etc. It also depends to some extent on the type of input signal. If the input contains a lot of random noise, a significant averaging time may be needed (to collect adequate data points) to produce an accurate estimate. [Fraden, 2004] provides a description of various such random effects (Type A) and systematic effects (Type B) that dictate the resultant accuracy.

[33] The quality of the calibration standard used also has a role to play in determining the accuracy. For instance, if the only available calibration standard is 10 times the resolution of the sensor, then it is not possible to obtain accuracy better than that.

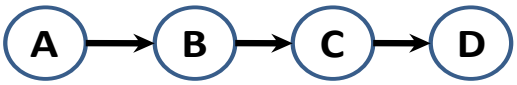
the belief values for a particular sensor or a process). The following section examines the effect of sensor accuracy in particular on the effectiveness of the proposed algorithm.

Range	State
1.0≤A≤1.5	1.25
1.5≤A≤2.0	1.75
2.0≤A≤2.5	2.25
2.5≤A≤3.0	2.75
3.0≤A≤3.5	3.25
3.5≤A≤4.0	3.75
4.0≤A≤4.5	4.25
4.5≤A≤5.0	4.75

Range	State
2.0≤B≤3.0	2.5
3.0≤B≤4.0	3.5
4.0≤B≤5.0	4.5
5.0≤B≤6.0	5.5
6.0≤B≤7.0	6.5
7.0≤B≤8.0	7.5
8.0≤B≤9.0	8.5
9.0≤B≤10.0	9.5

Range	State
1.0≤C≤1.5	1.25
1.5≤C≤2.0	1.75
2.0≤C≤2.5	2.25
2.5≤C≤3.0	2.75
3.0≤C≤3.5	3.25
3.5≤C≤4.0	3.75
4.0≤C≤4.5	4.25
4.5≤C≤5.0	4.75

Range	State
1.2≤D≤1.8	1.50
1.8≤D≤2.4	2.10
2.4≤D≤3.0	2.70
3.0≤D≤3.6	3.30
3.6≤D≤4.2	3.90
4.2≤D≤4.8	4.50
4.8≤D≤5.4	5.10
5.4≤D≤6.0	5.70



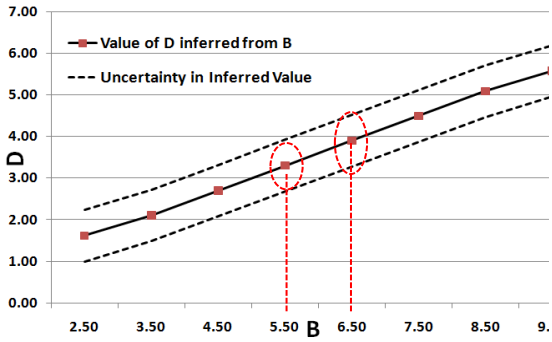


Figure 5-24: Effect of Sensor Characteristics on Inferencing

Consider the network in Figure 5-24. Suppose the ranges of the physical variables represented by the nodes *A*, *B*, *C*, and *D* are [1, 5], [2, 10], [1, 5] and [1.2, 6] respectively[34]. Each node is assumed to be discretized into eight states, based on the acceptable range of variation in each of the node states. The instantiation table for this network is the same as the one shown in Table 5-2. Considering the step where node *B* is the *NI* and node *D* is the *NoI*, the inferred value of *D* is obtained by setting the state of *B,* corresponding to the reading indicated by the sensor for *B* as evidence to the network.

[34] For this hypothetical example, it is assumed that the variables can be related by simple analytical relationships of the form $B=2A$, $C= 0.5\ B$ and $D= 1.2C$. Taking into account the intermediate process uncertainties, the relations may be written as $B=2A+ \varepsilon_{A->B}$, $C= 0.5B+ \varepsilon_{B->C}$, $D= 1.2C+ \varepsilon_{C->D}$; where the values $\varepsilon_{A->B}$, $\varepsilon_{B->C}$ and $\varepsilon_{C->D}$ represent the uncertainty in the processes *A->B, B->C* and *C->D* respectively. The conditional probability distributions are generated using a standard normal distribution centered on the states with a standard deviation of 0.15, 0.15 and 0.3 respectively.

Suppose there are two possible sensor options that may be used for measuring B, say B_1 and B_2 both of which have good resolution based on the bin sizes into which the probability distribution of B has been discretized (i.e. both sensors can measure any incremental change in the value of the variable B that falls within the individual bin limits) but the B_1 sensor has an accuracy of ±0.1 and the B_2 sensor has an accuracy of ±0.3 (both accuracy specifications refer to absolute values in measurand units).

Suppose the actual value of the physical parameter corresponding to B is 5.8. Ideally, this reading would correspond to the bin $5.0 \leq B \leq 6.0$ and to infer the value of D, the state B=5.5 should be instantiated in the network. If the sensor B_1 is used, then the actual output of the sensor based on which this instantiation would be done can range anywhere from 5.7 to 5.9, based on its accuracy. These values still fall within the bin $5.0 \leq B \leq 6.0$ and hence, the correct state of B can be instantiated. In this case, instantiating B=5.5 results in a value of D=3.3, as expected. However, with the sensor B_2, the sensor output may range from 5.5 to 6.1. Hence there is the possibility that any output greater than 6.0 from the sensor B_2 would lead to the state B=6.5 (corresponding to the bin $6.0 \leq B \leq 7.0$) to be instantiated, which in turn would lead to a value of D=3.9 as shown. Thus, the accuracy of the sensor corresponding to the instantiated node is crucial and must be taken into account along with the other application requirements when a sensor is being selected or designed for the system.

The above requirement may be stated as a rule of thumb as follows: 'The reading from a sensor may be used to instantiate a particular node state only if the possible range of variation in the value (calculated using the relation actual reading ± sensor accuracy) lies within the bin limits of the state that is being considered for instantiation.' In other words, if SR_U and SR_L represent the possible range of variation in the sensor reading after accounting for accuracy and if B_U and B_L represent the bin limits of the node state being considered for instantiation, then $B_L < [SR_U , SR_L] < B_U$.

The significance of the accuracy of the sensor corresponding to the node of interest can also be similarly illustrated. Suppose there are two sensors D_1 and D_2 that may be used to measure the variable D (with sufficient resolutions to measure any

incremental changes within any of the bins into which the probability distribution of D has been discretized) and accuracies of ±0.1 and ±0.2 for D_1 and D_2 respectively. Based on the presumed relations, a true value of B= 5.8 should result in a true value of D= 3.48. If the sensor D_1 is used to measure the value of D, then its actual output can range anywhere from 3.38 to 3.58, based on its accuracy. This would still correspond to the state D = 3.3 (bin 3.0≤ B ≤ 3.6) which is the value (most probable state) obtained via inferencing. Thus the measured and the inferred values can be considered to be in agreement in this case, which would reinforce the beliefs in the sensor for D and the associated processes B->C and C->D in the instantiation table.

If however, the sensor D_2 is used to measure the value of D, then its actual output can range anywhere from 3.28 to 3.68. This means that there is a possibility that the output of the sensor may be considered as corresponding to the state D =3.9 (bin 3.6≤ B ≤ 4.2). In this case, the measured and the inferred values would be considered to be conflicting, reducing the belief values for the sensor and the processes B->C and C->D in the instantiation table. Such situations may be avoided by careful selection of sensors with suitable accuracy based on the application requirements (or using redundant sensors with very high accuracy when possible).

Based on the accuracy of the different sensors corresponding to the different nodes in the network, that are actually integrated into the system and are responsible for monitoring it, various rules of thumb may be created to determine whether or not the reading from a particular sensor can be used to set a state of its corresponding node as evidence, or whether or not the reading from a sensor can be considered to be concurring with the inferred value for that measurand, during the fault detection and isolation algorithm implemented when the system is in operation.

For instance, in the example discussed, one such rule may be as follows: if the sensor B_1 is being used and if the actual value of the measurand B is between 5.9 and 6.1 (considering a band of readings that correspond to the accuracy of sensor B_1 and centered on the common bin limit 6.0 for the two adjacent states of the node B), a prudent choice

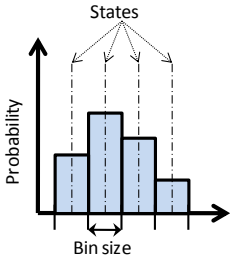
would be to avoid instantiating any evidence to the network with values that are indicated by the sensor within this range[35].

5.7.4. Effect of Discretization on Choice of Thresholds

In the fault detection and isolation procedure, the reading from the sensor corresponding to the *NI* is used to set a particular state of that node closest[36] to the measured value, as evidence to the network. The value indicated by the sensor corresponding to the *NoI* is then compared to the value that is obtained for the same node via probabilistic propagation and a binary decision is made as to whether these two values are in agreement or not. In order to determine which state of the *NoI* to associate with the sensor reading for comparison purposes, there is a need to define specific thresholds within which the sensor reading values would be considered as acceptable.

State	Prob.
A = 0.25 => p(0 ≤ A ≤0.5)	0.2
A = 0.75 => p(0.5≤ A ≤1.0)	0.37
A =1.25 => p(1.0≤ A ≤1.5)	0.295
A = 1.75 => p(1.5≤ A ≤2.0)	0.135

(a)



State	Prob.
A = 0.125 => p(0≤ A ≤0.25)	0.075
A = 0.375 => p(0.25≤ A ≤0.5)	0.125
A = 0.625 => p(0.5≤ A ≤0.75)	0.22
A = 0.875 => p(0.75≤ A ≤1)	0.15
A = 1.25 => p(1≤ A ≤1.25)	0.215
A = 1.375 => p(1.25≤ A ≤1.5)	0.08
A = 1.625 => p(1.5≤ A ≤1.75)	0.09
A = 1.875 => p(1.75≤ A ≤2.0)	0.045

(b)

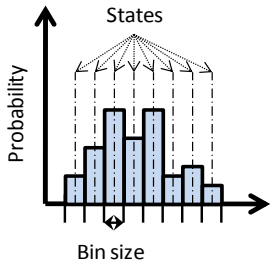


Figure 5-25: Effect of Discretization of States

[35] If the duty cycle of operation for the system is such that the value for the variable *B* lingers around 5.8 for a significant range of operation, such a rule might lead to a situation where the readings from this sensor are continuously rejected. This, in turn, may lead to false alarms from the SPFDI algorithm. If the readings are not used to update the performance maps (due to the accuracy restrictions), it may also have ramifications on the higher level CBM algorithms. Thus, it is also critical to take into account the expected range of values that a sensor is expected to encounter, in addition to its accuracy, while framing such rules.

[36] For instance, by calculating the difference between the value indicated by the sensor and the different state values of the instantiated node

The data available for each node/measurand is discretized into a number of states based on the application requirements. The acceptable upper and lower limits of variation in the node states are used to determine the sizes of the bins into which the values can be discretized. The smallest possible bin size that can be achieved is typically dictated by the accuracy of the sensor corresponding to that node (for example, [Mehranbod, et. al., 2003] uses a bin size which is twice the sensor accuracy). As a general rule, the bins can be defined centered on the state values with the size defined as the sensor accuracy multiplied by some factor (for instance, $\pm 3\sigma$, where σ is the accuracy of the sensor corresponding to that node).

Consider, for example, a random variable A, which can take the values in the range [0, 2] i.e. $0 \leq A \leq 2$. Figure 5-25 shows two possible ways in which A can be discretized with different number of states (*Note:* The states and probability values used here are purely for illustrative purposes). It can be seen that with a smaller number of states/larger bin sizes (as shown in Figure 5-25 (a)), there is a potential for a larger range of values of A falling under the same bin, which leads to a more skewed and a narrower distribution of the probabilities of each state occurring.

As a result, there will be certain states with a distinctly higher value of probability than the others. This allows the use of the bin limits corresponding to the state with the highest probability value (or the 'most probable state') as the threshold limits for a particular node to help to slot the reading indicated by the sensor corresponding to that node into one of the discretized states and enable comparison between measured and inferred values for that node. For instance, suppose that A is the *NoI* in a particular row of the instantiation table and by instantiating other nodes in the network, the node state A= 0.75 is obtained as the most probable state. The bin corresponding to this state of A is given by $0.5 \leq A \leq 1.0$. Thus, the values 0.5 and 1.0 which are the bin limits will be used as thresholds. Now, if the sensor corresponding to A indicates a reading of $A_{measured}$ =0.85, this value would be considered acceptable (i.e. $A_{measured} \equiv A_{inferred}$) since it falls within the limits of the bin which corresponds to the most probable node state.

When the values of A are discretized into a greater number of states, it can be seen that the probability values are also more evenly distributed across the different states as shown in Figure 5-25 (b). In such cases, often it becomes difficult to isolate a single state of the node, whose probability value is distinctly higher than those of all the other node states. For instance, the probability values for the states A=0.625 and A=1.25 are very close to each other. With any additional variations in these probability values that are introduced for instance due to rounding errors or due to approximate inferencing algorithms, it may not be possible to accurately determine the most probable state and hence the threshold values based on it. In such cases, some alternatives might be to:

a) Calculate the expected value of the posterior distribution as well as its variance or standard deviation, and use these values to determine the thresholds. For instance, the threshold limits may be specified in the form of $\mu \pm n*\sigma$, n=1, 2, 3 where μ is the mean and σ is the standard deviation

b) Add the bin size to the expected value. For instance, in Figure 5-25 (b), $E(A)$=0.91. So, the thresholds may be set based on the bin size of 0.25 as [0.785, 1.035]. Any sensor reading that falls within these limits would then be considered acceptable.

c) Specify a certain number of bins centered around the expected value (this would typically result in generous thresholds). For instance, in Figure 5-25 (b), the expected value $E(A)$=0.91 falls in the bin $0.75 \leq A \leq 1$. Choosing the two adjacent bins as threshold limits would correspond to the bins $0.75 \leq A \leq 1.0$ and $1 \leq A \leq 1.25$. Thus, if the sensor corresponding to A indicates a reading of $A_{measured}$ =1.1, this value would be considered acceptable (i.e. $A_{measured} \equiv A_{inferred}$) since it falls within the threshold limit.

Again, the question of how restrictive the thresholds need to be is dictated primarily by the application requirements.

5.8. USING W_S AND W_P VALUES IN LEARNING THE MODEL PARAMETERS

When the Bayesian network for a system is initially constructed, the model parameters (conditional probability distribution entries in the CPTs) for the different nodes are determined based either on an expert's opinion regarding how the system is likely to behave under different scenarios or they are based on data collected from extensively testing the system over its entire operational range (i.e. from the performance maps). In order to represent the status of the system at any instant as accurately as possible there is a need to refresh or update the CPT values (and hence the corresponding maps) with fresh sensor data on a periodic basis. This is referred to as 'learning' the model parameters [Cohen, et. al, 2001]. However, care must be taken to ensure that:

a) The data used to update the CPTs is reliable enough, so as to avoid corrupting the existing information, and

b) The process of updating is quick enough to accurately represent the prevalent system conditions, without the need to wait for a large number of data samples.

The SPFDI procedure described in the earlier sections provides valuable information in the form of beliefs (W_S and W_P) as to how trustworthy the sensors are as well as an indication of the health/status of the different components of the monitored system. These beliefs can now be used to determine if a particular sensor can be used to update the model parameters and also the rate at which the parameters of the various conditional probability tables in the network can be updated. The following section provides an adaptation of the online learning algorithm initially presented by [Bauer, 1997] and later refined by [Cohen, et. al, 2001].

The notation used in the following discussion is similar to that of [Bauer, 1997] and [Cohen, et. al., 2001]. Let X_i be any node in the network consisting of N nodes that can take the states $\{x^1, x^2, \ldots. x^k\}$. Let the different configurations that the set of parents PA_i of the node X_i can assume[37] be represented as $\{pa_i^1, pa_i^2 \ldots .. pa_i^j\}$. Then, every value in the CPT of the node X_i can be represented succinctly using the notation:

[37] Each configuration represents a particular column in cpt of the node X_i

$$\theta_{ijk} = P(X_i = x_i^k | PA_i = pa_i^j) \quad \text{Eq. (5-3)}$$

The objective is to update the value of θ_{ijk}^T for the CPTs of the different nodes at a given sampling instant *T*, to a new value θ_{ijk}^{T+1}, based on the results of an isolation cycle (when the beliefs in the sensors and the processes have been calculated) and a new set of samples is obtained $D_{T+1} = \{x_1, x_2 \ldots . x_N\}$, to accurately represent the updated system conditions that would be used in the next isolation cycle. Note that, in this case it is assumed that all the nodes are observable i.e. there is a sensor or a source of information associated with every node and there is no missing data i.e. at every sampling instant the value for each node in the network is available from its corresponding sensor.[38]

[Bauer, 1997] describes the problem of updating of the network parameters as a maximization problem that takes the form

$$\theta_{ijk}^{T+1} = argmax_\theta [\eta L_D(\theta) - d(\theta, \theta_T)] \quad \text{Eq. (5-4)}$$

where $L_D(\theta)$ represents the normalized log likelihood of the data given the network, *d* is the distance between the models at the two sampling instants and η is the learning rate. [Bauer, et. al. 1997] use the χ^2 distance and provide a solution to the maximization problem under the constraint that $\sum \theta_{ijk} = 1 \; \forall \; i, j$ and refer to it as the EM (η) algorithm.

[Cohen, et. al, 2001] provided an extension to this algorithm, termed the Voting EM algorithm for the online learning case. The authors state that in online learning, each new data sample represents a single instance of the network and present the following rule for updating the network parameters when all the nodes are observed:

$$\theta_{ijk}^{T+1} = \begin{cases} \eta + (1-\eta)\theta_{ijk}^T & \text{for } x_i = x_i^k \text{ in } D_{T+1} \text{ i.e. } P(x_i^k | D_{T+1}) = 1 \text{ and } P(pa_i^j | D_{T+1}) = 1 \\ (1-\eta)\theta_{ijk}^T & \text{for } x_i \neq x_i^k \text{ in } D_{T+1} \text{ i.e. } P(x_i^k | D_{T+1}) = 0 \text{ and } P(pa_i^j | D_{T+1}) = 1 \\ \theta_{ijk}^T & \text{otherwise} \end{cases} \quad \text{Eq. (5-5)}$$

[38] The situations with hidden nodes, or cases where a node X_i is observed and its parents are not observed or vice-versa are not considered. The approach to learning in these cases can be found in the work of [Lim and Cho, 2006], [Zhang, et. al, 2003]

With each new sample of data, if the observed values of the child and the parent nodes are in an appropriate configuration, it represents an increase in the confidence for the observed state of the child node and the corresponding importance of that state in the overall distribution is increased (while the importance of the other states are decreased by the same amount) by a value that is proportional to the learning rate η. [Cohen et. al, 2001] provide an explanation for the update rule in Eq. (5-5) as follows: when a new sample of data D is available for all the nodes in the network and it is observed that the parents of a node X_i are in the j^{th} configuration and if, for this configuration, the corresponding value of X_i should be its k^{th} value, and if the observed value of X_i in D is indeed x_i^k, then increase the value of θ_{ijk}. If it is observed that the parents of X_i are in the j^{th} configuration but the corresponding observed value of X_i in D is not x_i^k, then decrease the present value of θ_{ijk}. Finally, if the parents of X_i are not observed to be in the j^{th} configuration in D, then the value of θ_{ijk} remains unchanged. As more and more data samples are analyzed, the estimated probabilities converge to their true values.

The value of η determines the amount by which the past data is weighted in order to update the parameters. As the value of η approaches zero, the value of past data is weighted significantly and the model parameters remain practically unchanged. With a smaller value of η, the rate of convergence is slower but smoother and not very sensitive to the new sample of data; the final variance (a finite, non-zero value) of the estimate is also smaller [Cohen et. al, 2001]. Conversely, as the value of η approaches 1, the newly available or present data is assigned a higher importance in determining the updated value of the parameters. A value of η=1 provides the fastest convergence, but also results in a high variance [Cohen et. al, 2001][39]. Thus, the value of η affects the rate of convergence as well as the variance in the final estimated value. Although a variance around the estimated probability value may seem undesirable, [Cohen et. al, 2001] state that it allows the algorithm to adapt to changing environments (in conjunction with the

[39] The magnitude of variance also depends on the true probability value θ_{ijk} and decreases as it approaches 1 or 0. [Cohen, et.al, 2001] demonstrate that the variance is highest for a true probability value of 0.5

appropriate values) as well as escape from local maxima (when the estimated probability converges towards the true value) when new evidence becomes available.

The work of [Cohen, et.al. 2001] indicates the use of a fixed learning rate η and the use of the same η for all the network parameters as the shortcomings of the Voting EM algorithm. Using the same η for all the network parameters results in good estimates for certain values but may perform poorly for others. Hence, the authors devised a modified version of the Voting EM algorithm using an adaptive learning rate, with different learning rates for the different columns of the CPT for a given node (similar to the maximum likelihood estimation for CPT entries as described by the authors). Starting with an initially high value, the value of η is reduced when the estimated parameter value is close to convergence and is increased when there is a large error between an estimated value and its mean [Cohen, et.al. 2001]. This change allows the algorithm to adapt to changes in the network parameters (due to a change in the modeled environment) as well as break out of any local maxima during the estimation process. For the present discussion, let the adaptive learning rate for a node X_i whose parents are in the j^{th} configuration be represented as $\eta_{PAj,Xi}$ i.e. the value of η used in the set of equations in Eq. (5-5) is unique to every column (corresponding to PA_j) in the CPT of node X_i. The application of these rules to update the node CPTs can now be explained using a simple example.

Consider a system consisting of four variables of interest and represented by the Bayesian network in Figure 5-26. Suppose the range of the physical variables represented by the nodes *A*, *B*, *C*, and *D* are [0, 4], [0, 8] [0, 4] and [0, 12] respectively. For this hypothetical example, it is assumed that the variables can be related by simple analytical relationships of the form $B=2A+\varepsilon_{A->B}$, $C=0.5B+\varepsilon_{B->C}$, $D=3C+\varepsilon_{C->D}$; where $\varepsilon_{A->B}$, $\varepsilon_{B->C}$ and $\varepsilon_{C->D}$ represent the uncertainty in the processes *A*->*B*, *B*->*C* and *C*->*D* respectively. Each of the nodes is discretized into four states each[40]. The discretization, the conditional probability tables for these nodes and the learning rates for the three processes $\eta_{Aj,B}$, $\eta_{Bj,C}$

[40] The discretization of states for the distribution of node *A* represents $p(0\leq A\leq 1)=0.25$, $p(1\leq A\leq 2)=0.25$, and so on. A similar explanation is valid for the other nodes as well

and $\eta_{Cj,D}$ are also shown in Figure 5-26. The subscript *j* for the learning rates denotes the configuration of the parent node for that process.

A=0.5	0.25
A=1.5	0.25
A=2.5	0.25
A=3.5	0.25

	A=0.5	**A=1.5**	**A=2.5**	**A=3.5**
B=1	0.9	0.05	0.0	0.0
B=3	0.1	0.9	0.05	0.0
B=5	0.0	0.05	0.9	0.1
B=7	0.0	0.0	0.05	0.9

	B=1	**B=3**	**B=5**	**B=7**
C=0.5	0.8	0.1	0.0	0.0
C=1.5	0.2	0.8	0.1	0.0
C=2.5	0.0	0.1	0.8	0.2
C=3.5	0.0	0.0	0.1	0.8

	C=0.5	**C=1.5**	**C=2.5**	**C=3.5**
D=1.5	0.7	0.2	0.0	0.0
D=4.5	0.3	0.6	0.2	0.0
D=7.5	0.0	0.2	0.6	0.3
D=10.5	0.0	0.0	0.2	0.7

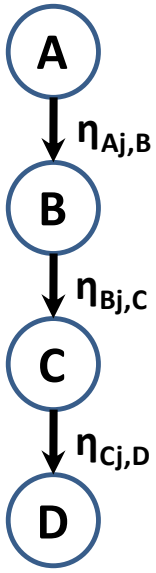


Figure 5-26: Learning Rates for Processes in the Network

Suppose at a sampling instant *T*, the values indicated by the sensors for the nodes *A*, *B*, *C*, and *D* are *A*=1.8, *B*=3.4, *C*= 1.6 and *D* =4.8 respectively. Now, consider only the process *B*->*C*. Based on the readings from the sensors for nodes *B* and *C* it can be determined at the end of the preceding fault detection and isolation cycle that the process *B*->*C* and the sensors for *B* and *C* are not faulty. Thus, for the parent configuration of *B*=3, the corresponding observed value of the child node *C* should be *C*=1.5. Based on the discretization of the nodes as shown in the CPTs in Figure 5-26, this is indeed the case. Hence the value of θ_{ijk} corresponding to *C*=1.5 in the CPT of node *C* needs to be increased and the values of θ_{ijk} for all the other states of *C* need to be decreased. For a value of *B*=3.4, the learning rate to be used is the value of η_B corresponding to the column *B*=3 in the CPT for node *C* i.e. $\eta_{B=3,\ C}$. The change in the CPT values for values of node

C, by applying the rules in Eq. (5-5), for three distinct values of $\eta_{Bj,C}$ =0, 0.5 and 1 are shown in Figure 5-27[41].

	B=1	B=3	B=5	B=7
C=0.5	0.8	0.1	0.0	0.0
C=1.5	0.2	0.8	0.1	0
C=2.5	0.0	0.1	0.8	0.2
C=3.5	0.0	0.0	0.1	0.8

	B=1	B=3	B=5	B=7
C=0.5	0.8	0.1	0.0	0.0
C=1.5	0.2	0.8	0.1	0.0
C=2.5	0.0	0.1	0.8	0.2
C=3.5	0.0	0.0	0.1	0.8

	B=1	B=3	B=5	B=7
C=0.5	0.8	0.1	0.0	0.0
C=1.5	0.2	0.8	0.1	0.0
C=2.5	0.0	0.1	0.8	0.2
C=3.5	0.0	0.0	0.1	0.8

$\eta_{B=3,C} = 0$

	B=1	B=3	B=5	B=7
C=0.5	0.8	0.05	0.0	0.0
C=1.5	0.2	0.9	0.1	0.0
C=2.5	0.0	0.05	0.8	0.2
C=3.5	0.0	0.0	0.1	0.8

$\eta_{B=3,C} = 0.5$

	B=1	B=3	B=5	B=7
C=0.5	0.8	0.0	0.0	0.0
C=1.5	0.2	1.0	0.1	0.0
C=2.5	0.0	0.0	0.8	0.2
C=3.5	0.0	0.0	0.1	0.8

$\eta_{B=3,C} = 1.0$

Figure 5-27: Effect of Different Learning Rates

As mentioned earlier, η controls the rate of convergence and the extent to which past or newly available data is relied upon. It can be seen that for a value of $\eta_{B=3,C}$ =0, there is no change in the probability distribution of *C*; i.e., there is no learning; the past data is relied upon completely and the newly available data is not used in updating the parameters of *C*. When $\eta_{B=3,C}$ =1, it can be seen that the column corresponding to *B*=3 in the CPT of node *C* has now changed, indicating that the newly available data has been completely relied upon in updating the parameters of *C* as compared to past data, which is completely forgotten[42]. For all other intermediate values, faster convergence is obtained with the learning process with the values of η closer to 1, and conversely slower convergence with values of η closer to 0. The choice of an appropriate value for η is thus crucial. The following section explores how the beliefs in the different sensors and processes in the network can be used to determine the values of η and provides some guidelines.

[41] $\eta_{Bj,C}$ =0 ∀ j implies that $\eta_{B=1,C}$ =0, $\eta_{B=3,C}$ =0, and so on.

[42] As mentioned in [Cohen, et. al, 2001], for a value of η=1, the estimated probability oscillates between 0 and 1 based on whether the value for the node X_i is equal to its k^{th} value in the sample of data being analyzed

In order to set the learning rate adaptively, certain factors need to be considered:

a) The starting value of η for all the nodes in the network needs to be known.

b) When a new sample of data is obtained and analyzed, a decision has to be made whether the learning rate has to be increased, decreased or maintained the same.

c) Finally, the magnitude by which the learning rate is increased or decreased also needs to be determined.

The initial values of $\eta_{PAj,Xi}$ for the different nodes may be determined by the user based on the number of samples that have been analyzed previously for a given node X_i, for each of its different parent configurations PA_j. For instance, if the system is new and relatively untested, there is inadequate information to estimate the CPT parameter values that would represent that node accurately. In such cases, it would be desirable to have a high value of η (closer to 1, say 0.8) for faster convergence to estimate the parameters as quickly as possible and gradually reduce the rate as more and more data becomes available, which in turn would help reduce the variance in the estimated values. On the other hand, if the system has been operational for a while, then it is likely that a large quantum of data samples is already available that can be used to estimate the CPT parameters quite accurately. In such a case, a low starting value of η might be sufficient (closer to 0, say 0.01?). The number of samples based on which this judgment is made is highly application-specific. For a system that does not change very quickly, a lower number of samples (say, 1000) may be sufficient to estimate the CPT parameters accurately, but for a highly dynamic system or for a system that degrades quickly, a large number of samples (say, 10000) might be needed in order to make an accurate estimation of the CPT parameters for the various nodes.

In order to decide when and by how much to modify the learning rate, [Cohen et. al., 2001] use the mean and the variance of the estimated parameters in conjunction with user-defined thresholds as the decision criteria and an exponential increase/decrease in the learning rate respectively. In this research, the use of the values of beliefs in the sensors for each of the nodes in the network, obtained at the end of the fault isolation cycle, is proposed both as the decision criterion to modify the learning rate and also to

determine the magnitude of this change. In the situation that all the sensors and processes are found to be operating correctly (as indicated by W_S and W_P values which are ~1 or above certain user-defined thresholds at the end of the fault isolation cycle), the learning rates need to be adjusted to update the parameters based on the data sample analyzed. Let $\eta^L_{PA_jX_i}$ and $\eta^H_{PA_jX_i}$ be the lowest and the highest learning rates respectively for a particular combination of node value X_i and its parents in the configuration PA_j. If NS is the number of data samples that have been previously analyzed to determine the learning rate, the new value of $\eta_{PAj,Xi}$ for the next learning cycle can be calculated as follows:

$$\eta_{PA_jX_i} = \eta^L_{PA_jX_i} + \frac{\eta^H_{PA_jX_i} - \eta^L_{PA_jX_i}}{1 + NS} \qquad \text{Eq.(5-6)}$$

The very first time there is a modification in the learning rate, the value of NS is zero and the learning rate is set to its highest value. Note that as the number of data samples increases, the learning rate decreases, theoretically attaining a value of $\eta^L_{PA_jX_i}$ (or zero if $\eta^L_{PA_jX_i}$= 0) after an infinite number of samples have been analyzed. Practically, the learning rate keeps decreasing but remains a finite value (similar to the heuristic used in [Cohen, et. al., 2001]).

If, however, the sensor corresponding to a particular node X_i in the network is identified as being potentially faulty at the end of the fault isolation cycle preceding the latest data sample (indicated by its W_S value approaching a value of zero or lesser than a user-defined threshold), then

a) The $\eta_{PAj,Xi}$ for all the columns in the CPT of that node is set to a value of zero immediately.

b) Since the values indicated by this sensor determine the parent configurations PA_r for all the child nodes of X_i, say Y_k, the learning rates for all those nodes, $\eta_{PAr,Yk,}$, are also set to zero immediately.

This is done to prevent the corruption of the existing parameter values in the CPT of X_i, in the situation that the corresponding sensor is actually faulty. If however, after subsequent fault isolation cycles, it turns out that the sensor is not faulty (or if a faulty

sensor has been replaced with one that is operating correctly), the learning rate may be reset to a value that was used just preceding the fault isolation cycle in which an alarm for the faulty sensor was first raised.

Now consider a situation where at the end of the fault isolation cycle it is determined that all the sensors are operating correctly (all the W_S values are ~1 or above certain user-defined thresholds) but there is a process fault (one or more W_P values are ~0 or below certain user-defined thresholds). In this case, if the learning rate is too low, then the CPT parameters cannot be updated quickly enough to represent the variation in the system dynamics. Hence, the learning rate needs to be increased. In order to do this, once again the W_S and W_p values can be used to determine the magnitude by which the learning rates can be changed. Suppose $PA_j \equiv \{P_1^a, P_2^b, \ldots P_m^q\}$ represent the particular configuration of the parents P_1, P_2, $\ldots P_m$ of the node X_i, and there is a fault in the k^{th} process P_k-> X_i. Once all the sensors have been deemed trustworthy based on their W_s values, the new increased learning rate for the particular combination of PA_j and X_i may be calculated as follows:

$$\eta_{PA_jX_i}^{new} = \eta_{PA_jX_i}^{current} + \left(1 - W_{P_{avg}}\right)\left(\eta_{PA_jX_i}^{H} - \eta_{PA_jX_i}^{current}\right) \qquad \text{Eq.(5-7)}$$

where $W_{P_{avg}}$ is the average of the W_P values obtained by considering all the processes, namely, P_1-> X_i, P_2-> X_i,.. P_m-> X_i that terminate at X_i. It can be seen that the new value of the learning rate is determined by the condition of the system. Since all the sensors are deemed to be operating correctly, the faultier the system is (indicated by low W_P values for one or more processes), the higher is the learning rate. This is desirable since it will help update the parameters quickly and can help improve the output from the higher level condition-based maintenance algorithms. For instance, if at the end of an fault isolation cycle it is determined that all the processes that terminate at the node X_i are operating correctly, and all the sensors are also operating normally, it was observed earlier that all the W_S and W_P values attain a value of 1. From the above equation, it can be seen that in this situation, the learning rate remains unchanged (as $W_{P_{avg}}$ is also 1). In the extreme scenario when it is known that all the processes are faulty ($W_{P_{avg}} \approx 0$), then the learning

rate is set to its highest value. For any other intermediate condition ($0< W_{P_{avg}} \leq 1$), the learning rate is increased from its present value[43].

5.9. CHAPTER SUMMARY

This chapter presented the development of a novel Bayesian network based algorithm for detection and isolation of sensor faults, and distinguishing sensor faults from process faults. Using illustrative examples (including the test set-up described in Appendix B), the performance of the proposed Sensor and Process Fault Detection and Isolation (SPFDI) algorithm was demonstrated by considering different types of faults. These demonstrations showed that the proposed method has the following advantages:

- The proposed method complements the overall Bayesian network based decision making framework for intelligent systems developed in [Ashok and Tesar, 2007]. Thus, there is no need for multiple/alternative methodologies to exist in parallel in order to achieve different objectives like system modeling, data validation, etc.
- The same Bayesian network structure used for decision-making at the system level (for instance, combining appropriate performance maps into performance envelopes) can also be used to examine the validity of the readings obtained from the sensors that monitor the system. This obviates the need for creating multiple networks to address different objectives like detection of faults, isolation of fault sources, etc.
- The algorithm does not require the existence of explicit closed form relationships between the different variables of interest. Fault detection and isolation is performed solely on the basis of the knowledge of how the system is likely to behave as indicated by the performance map data (and encoded in the CPTs of the different variables) and the data obtained from all the sensors monitoring the system at a given instant.

[43] In the discussion above, the probabilities of false alarms (Section 5.7.1) are not taken into account while updating the model parameters. The modification of Eq.(5-6)and Eq.(5-7) in order to accommodate those probabilities is a topic of future research.

- The algorithm also results in a relatively simple network structure. Since explicit nodes modeling the health of the different system components (processes) or the sensors are not required, the total number of nodes/size of the network is reduced[44].
- The algorithm does not require apriori knowledge of all the fault modes and their effect on the behavior of the different sensors and process in the system.
- The algorithm does not require the assumption of the system condition remaining unchanged (i.e. no degradation in system health) in order to detect and isolate faulty sensors or vice-versa. This is a significant advantage over many other approaches (Section 5.3.3), as it enables the algorithm to distinguish between potential faults that occur in the sensors versus those that may occur in one of the system components.
- The algorithm can handle single as well as multiple faults in sensors as well as processes, to the extent permissible by the subset issue (Section 5.5.3).
- The belief values for the sensors and process *(W_S* and *W_P*) provide an intuitive indication regarding the health of any sensor or process in the system, with low values indicating potential faults and high values indicating regular operation. These belief values can be interpreted easily by the system operator to make decisions regarding which sensors to use to achieve the system's performance goals in the best possible manner or whether or not to use certain sensors to update the stored maps/decision surfaces, etc.
- The algorithm is an any-time algorithm i.e. beginning with an assumption of no knowledge regarding the condition (faulty/operational) of a particular sensor or a process in the system, with each subsequent step in the instantiation table, the knowledge becomes more and more certain. The belief values obtained at the end of a single fault detection and isolation cycle, provide a clear indication of the potential source/s of variation in the sensor reading/s based on the sample of data

[44] This may also result in more compact CPTs (depending on the application) since the number of parent nodes corresponding to any node in the network is restricted.

available at that point (if the same results are obtained over multiple samples, then it is indicative of a definite fault in that particular sensor or process).

Table 5-40: Benefits of Sensor and Process Fault Detection and Isolation Algorithm

	Benefit	Description
1.	Complements overall decision-making framework for intelligent systems	The same Bayesian network used for decision making is also used for fault detection/isolation
2.	Obviates the need for multiple methodologies to exist in parallel	For system modeling, operational decision-making, fault detection/isolation, etc.
3.	Simpler network structures	Does not require explicit modeling of health of various components of the system
4.	No need for apriori knowledge of fault modes	Enables CPTs to be generated relatively easily with available data
5.	Purely data driven	Does not require existence of closed form equations between variables of interest
6.	No need for assumption of invariant system health	Enables detection of any issues in the system components in addition to detection and isolation of faulty sensors
7.	Can detect multiple faults in sensors and processes	The algorithm can reliably detect faults in multiple sensors and processes in the system (to the limit imposed by the subset issue in the instantiation table)
8.	Easy interpretation of W_S and W_P belief values for decision making	The range of values $0 \leq W_S, W_P \leq 1$ provides a clear indication of fault to the system operator
9.	Anytime algorithm	The knowledge of the status of a particular sensor or process (i.e. whether it is faulty/non-faulty) becomes more certain with time
10.	Applicable to variety of domains	Flexibility in the choice of thresholds, potential ability for fault classification, ability to perform both predictive and diagnostic inferencing, given the Bayesian network structure allows implementation of the algorithm with relative ease

- Although the issue of fault classification (into categories like drift, bias, precision degradation etc.) has not been addressed in the present work, the use of the most probable state as a comparison criterion for the node of interest may allow such

functionality (for instance, by tracking the most probable state continuously over multiple cycles of validation as proposed in [Mehranbod, et. al., 2003])

- Given a Bayesian network representation for a system (with sensors to measure all the nodes), the framework for the proposed algorithm (Section 5.4) allows great flexibility in the choice of thresholds, in the choice of nodes for instantiation (as *NI*) and comparison (as *NoI*), type of inferencing (predictive/diagnostic) to generate the instantiation table, in accommodating variables that vary on different scales (for instance, as shown in Section 5.6), etc. This implies that the algorithm may be potentially implemented with relative ease across a wide variety of application domains, a few of which are discussed in the following chapter.

Chapter 6. Applications

6.1. INTRODUCTION

The work presented thus far has primarily been focused on intelligent EMAs. However, the same concepts are applicable across a wide spectrum of other areas. The common defining characteristic of these applications is that they involve complex systems monitored by multiple heterogeneous sensors; where multiple performance objectives have to be met with human choices involved in the operational decision making process. The following sections examine the relevance of the work presented in the earlier chapters by considering some potential scenarios in four broad classes of applications. Section 6.2 briefly outlines the different application classes, with some representative applications for each class. This is followed by in-depth analysis of a selected application for each class in Sections 6.3-6.6, demonstrating the applicability of the Sensor and Process Fault Detection and Isolation (SPFDI) algorithm developed earlier.

6.2. CLASSES OF APPLICATIONS

The Sensor and Process Fault Detection and Isolation (SPFDI) algorithm developed in Chapter 5 (Section 5.4), was applied to a system comprised of a four phase switched reluctance motor, instrumented with current, voltage, torque, speed and acoustic noise sensors. The effectiveness of the algorithm in distinguishing between sensor and process faults as well as in isolating the potential sources of those faults was demonstrated by considering different combinations of single and multiple faults. The methodology demonstrated is however, general in nature and applicable to a wider range of application domains beyond intelligent EMAs. The main requirements to implement the algorithm for any system are the ability to develop an accurate Bayesian network representation of the system (based on the knowledge of domain experts or based on learning from a database of the variables of interest in the system) and the ability to measure the physical phenomena represented by each of the nodes in the network, in real-time during system operation.

The utility of the SPFDI algorithm can be better understood by examining its relevance to four broad classes of applications (Table 6-1).

Table 6-1: Four Classes of Applications

	Class	Example Applications
1	Human safety at stake	Aircraft subsystems
		Robotic surgery
		Automobiles
2	System availability	Battlefield robots
		Submarines
		Nuclear power plant
3	High cost of system failure	Manufacturing line
		Remotely operated vehicles
		Wind turbines (gearbox)
4	Performance maximization	Ground combat vehicle
		Serial /Parallel manufacturing cell
		War fighter monitoring

6.2.1. Human Safety at Stake

This class encompasses applications where the decision maker has to decide on a particular course of action for the overall system operation, based on an estimation of the system capability using the information obtained from the sensors monitoring it. In doing so, the decision maker has to determine the trustworthiness of the available information in order to determine the ramification of making a particular choice. This is critical in situations where a significant degradation in the performance capability of the system (in reality or perceived due to its misrepresentation by faulty sensors) may jeopardize the safety of the humans interacting with or utilizing the system. If all the sensors indicate the values as expected for their measurands under the extant operating conditions, the decision process becomes straightforward (the system is operating normally and there is no associated risk). However, if a particular sensor or a subset of sensors indicates values that deviate from the anticipated values (for instance, as obtained from the performance

maps for those measurands), the decision maker needs to be able to determine whether the values are indicative of faulty sensors or if it indicates a potential issue with the system. This can be achieved by applying the SPFDI algorithm. The belief values for the different sensors and processes obtained from the algorithm can not only help provide a clear guidance to the decision maker regarding the status of the different components of the system but also help determine what alternative data sources may be used if it is found that a particular sensor is faulty and needs to be taken out of service. The following paragraphs provide two example scenarios where the availability of such information can prove to be life-saving.

There has been an increased emphasis on all-electric systems in next generation aircraft designs, replacing conventional hydraulic actuation systems (used to drive control surfaces, landing gear, etc.) with electromechanical (EMA) or electrohydraulic actuators(EHA) [Brown, et. al, 2009] [Koran and Tesar, 2008] [Byington, et. al, 2004] due to their increased reliability, lower maintenance and overall life cycle costs. Actuation systems, especially those that operate the flight control surfaces (ailerons, flaps, elevators, stabilators, etc) are critical not only from the aircraft performance point of view [Koran and Tesar, 2008] but also from perspective of pilot safety and survivability. Hence the area of fault detection and isolation (detection, identification and accommodation of sensor and actuator failures) and fault tolerant flight control has been an active area of research [Heredia and Ollero, 2010][Brown, et. al, 2009][Hajiyev and Caliskan,2005][Kobayashi, 2003][Napolitano, et. al, 2000]. It is evident from such research that loss of control of such flight control actuators due to faulty sensors is as serious an issue as failure of the actuator itself[1].

Consider for instance, a linear EMA driving the stabilator in a fighter aircraft. The EMA should be capable of producing sufficient moment to overcome inertia as well as external load forces due to gravity and aerodynamics [Koran and Tesar, 2008]. Suppose the force exerted by the EMA is measured using a force sensor and in addition, current and linear position sensors (to determine the ram motion) are also used to monitor its

[1] Such critical flight control systems are often equipped with quadruplex hardware redundancy (in sensors and actuators)to avoid a catastrophic failure [Lo, et. al. 2009]

performance. If the pilot (or the flight control computer) notices an unusually high reading from the force sensor, then using a Bayesian network model of the system (for instance consisting of nodes like current, linear position, sensed force, EMA ram force, ram position, etc.) and the data from the three sensors, the SPFDI algorithm can help determine if the anomaly is due to a faulty force sensor or due to an increased load acting on the stabilator itself (for instance, due to high turbulence). In a combat scenario, if it can be determined that the high force reading is due to an inadequately performing stabilator EMA, it may provide the pilot valuable information to reconfigure the other flight control surfaces to increase the chances of survivability of the aircraft.

Minimally invasive surgery using robotic systems is another example pertinent to this class of applications. These procedures are becoming popular due to the benefits they offer- reduced pain, faster rehabilitation/recovery time, better cosmetic results, etc. Typically such systems consist of a master-slave arrangement of robotic arms, where the surgeon remotely controls the master system using consoles with haptic and visual feedback. The slave system consists of multi degree of freedom robotic arms which introduce slender surgical instruments into the human body through small incisions [Tesar, et. al, 2006][Camarillo, et. al, 2004][Deml, et. al., 2004]. The gross hand motions of the surgeon at the master controller (after filtering tremors) are translated to actuation commands for fine motions of the slave surgical robot using suitable kinematic model transformations.

In this process, with no direct tactile feedback, the haptic feedback experienced by the surgeon plays a dominant role in the surgeon's ability to determine the necessary forces required to be applied between the surgical instruments and the human body. To accurately determine the interaction forces between human tissue and the instruments, miniature force sensors are integrated into the instruments themselves [Kuebler, et. al, 2005]. The reading from these sensors is scaled and used by the master controller to provide the suitable haptic cues to the surgeon to enable them to determine the appropriate force/motion levels required to make the incision [Deml, et. al, 2004].

Consider a scenario where the surgeon experiences a significant force from the haptic feedback system. In such a situation, if the surgeon attempts to counteract the high feedback force with a significant instrument force there is a potential risk of trauma to the patient as well as damage to the delicate surgical instruments. Thus, it becomes imperative for the control system to determine if this behavior is due to a high friction/cutting force between the surgical instrument and the human tissue at the point of insertion (for instance, as described in [O'Leary et. al, 2003]) or if it is due to a biased force sensor in the instrument.

Using an appropriate Bayesian network representation of the entire system (potentially consisting of nodes like sensed force, interface friction, instrument angle, joint angles of the slave surgical robot, etc.) and the data from the various sensors in the system, the SPFDI algorithm can help the master controller determine if the anomaly is due to a faulty force sensor or due to an abnormal angle of entry of the surgical tool leading to the increased operating forces. If it is found that the force sensor is operating correctly, this information may be then used to send a warning signal to the surgeon who can then take the corrective action required to reduce the risk of injury to the patient.

6.2.2. System Availability

This class of applications encompasses situations where a decision has to be made by the decision maker whether or not the system in its present condition is capable of performing the tasks required of it, before being deployed. This is termed as 'availability' of the system. The decision is made based on the information available from the sensors monitoring the system following a start-up/system performance check process that puts the system through a series of selected operational maneuvers representative of its actual duty cycle (with the SPFDI algorithm also being executed simultaneously). The goal is to avoid a costly shutdown of the system once it is deployed (for instance, due to a false alarm) or a catastrophic complete failure in the worst case if some potential sources of single point component faults remain undetected before the system is put into operation.

Included under this category would be applications wherein there are multiple identical systems that are capable of carrying out the task at hand and a decision has to be

made regarding which is the best option available at hand (the system with the greatest availability) to accomplish those goals.

A typical example of such a situation can be found in a battlefield scenario where a soldier may have different configurations of mobile robots at his/her disposal for detecting and disposing of explosives. The soldier may run each of these robots through a system check process that may involve running the robots through a specific motion plan (for instance, like the ones described in [Kulkarni and Tesar, 2009]), in order to determine which of the robots is most suitable, in terms of its 'availability' to do the task.

During this process, the performance of each of the different subsystems in the robots may be monitored using the appropriate sensors. For instance, this may be done for the drive wheels using sensors like current, voltage, temperature, etc. integrated into the drive wheel actuators. If the current sensor for one of the drive wheels indicates an unusual value, the soldier may be able to interpret the results of the SPFDI algorithm and determine that the cause of the abnormal reading is a faulty sensor and the drive wheel actuator itself is not faulty and that the robot is still capable enough to accomplish its mission. If the robot is deployed with the faulty sensor in place, such knowledge can be factored in, while providing a performance level assessment of the system (i.e., a measure of its effective availability) while it is in operation and the other operational sensors may be used to infer the unavailable measurand by using them as evidence in Bayesian inferencing algorithms (Section 2.5.2).

Another example application in this class may include a system that has been dormant or shutdown for a period and a decision has to be made whether to bring it back into active service. A typical example may be a submarine or a ship that is in the docks for scheduled maintenance and the vessel commander has to make a decision based on a series of sea trials if the vessel can be deemed to be seaworthy enough to return to active service and accomplish its mission objectives. For instance, in a submarine, the initial trials may consist of determining the range of motion of all the control surfaces that are used to maneuver the vessel during actual missions. The generation of acoustic noise during such motion is an important criterion for the vessel to retain its stealth ability. If

during the initial trials it is observed that the acoustic noise levels are higher than normal, then using all the available sensor readings (from the sensors for various measurands like current, voltage, position, acoustic noise, etc. integrated into the actuators for these control surfaces [Tesar, et. al, 2010]) and implementing the SPFDI algorithm may help the vessel commander decide whether the abnormal readings are due to a faulty acoustic noise sensor or if the additional noise is being generated from one of the active control surfaces, the main bearing for which may be potentially damaged. Such knowledge, combined with the commander's experience can enable the accurate assessment of the risk involved in deploying the vessel with the faulty component (sensor, bearing or actuator) or indicate a clear need to replace it.

6.2.3. High Cost of System Failure

This class encompasses applications that can be divided into two sub-categories. The first category includes applications where the decision maker (who is remotely monitoring and controlling the system) may decide to temporarily shut down a system and take remedial action, due to a perceived inability of the system to accomplish a desired task. However, such a decision could be the result of a wrong assessment of the system capability due to potential faults in one or more sensors monitoring it, when in reality the system is fully capable of performing its required task. The second category constitutes the other end of the spectrum, where the system capability is similarly misestimated due to faulty sensors and the system is made to operate well beyond its design limits for extended time. Doing so might cause irreparable damage to the system or complete catastrophic loss of the system in the extreme situation.

In either case, the effect of the inability to distinguish a potential sensor fault from a potential system fault would be tangibly felt in terms of its economic impact (for instance, in terms of a dollar value signifying the lost productivity from the system or the overall cost of the system itself). Thus, also in this class of applications, quantifying the dependability of the available information (via belief values for the sensors and processes using the SPFDI algorithm presented in Chapter 5) can help the decision maker evaluate the risk of making a particular choice.

A manufacturing line for food products can be considered as a good example of the first category in this class of applications. Given the need to manufacture high quality products while having to comply with several rigorous requirements for quality and safety, this application tends to be a sensor-rich environment. A variety of sensors are used for process control at different points along the manufacturing line to ensure a high quality end product.

For instance, [Neethirajan et. al., 2009] state that food packaging is often carried out in a modified atmosphere consisting of gases like CO_2 in order to prevent microbial food spoilage. A decrease in the concentration of CO_2 is taken as indicative of leakage which may affect the freshness of the end product. On the other hand, very high concentrations may also adversely affect the quality of the end product. Hence, various types of CO_2 sensors are used to accurately monitor the concentration of the gas used. The demand for high standards for food safety has also led to a proliferation of sensors used for monitoring food quality [Connolly and O'Reilly, 2005]. For instance, different types of sensors are used to detect the presence of pathogens, chemical toxins and pesticide residues in food as described in [Patel, 2002]. Other types of sensors could include like viscosity/ flow meters, density meters, temperature sensors, pH level sensors, etc.[Singh, et. al., 1997]; machine vision systems are commonly used during the final packaging step for ensuring consistency or grading of product size.

The sensors used in such food manufacturing applications are often subject to extreme conditions (factors like high pressure, temperature, etc. are common in many processes). Fouling problems are also sometimes observed if there is direct contact of the sensors with the food product. To remove the fouling, concentrated or heated solutions of chemicals are pumped through the system [Winquist, et. al., 2005] which may also affect some of the sensors used in the application. If the readings from such degraded sensors deviate drastically from the anticipated values the manufacturing plant supervisor may be inclined to halt production temporarily to ensure that product quality is not being hampered (for instance, if the reading from the CO_2 concentration sensor is high in the example cited earlier).

In such situations, time is often of essence since the raw materials for the food products need to be processed in a timely fashion to avert any degradation in their quality. A temporary shutdown of the production line can not only result in reduced output volume of the end product but can also result in a significant cost incurred due to the perishable nature of the raw materials. For instance, consider the pasteurization process used in the dairy food industry. Suppose the process utilizes a voltammetric electronic tongue or a taste sensor to monitor quality changes of milk (as described in [Winquist, et. al., 2005]) in addition to sensors that monitor the temperature and viscosity of milk. If the taste sensor indicates a very low reading (indicating that the quality of the milk is not up to the mark for it to be considered pasteurized to the acceptable degree) but the temperature and viscosity sensors indicate their expected values, then using a Bayesian network model of the system and the SPFDI algorithm, the plant supervisor can confirm that the fault lies with the taste sensor and continue to process the present batch of products until completion before deciding to shut down the line temporarily to replace the faulty sensor.

An example of the second category of applications in this class may be a system like an unmanned aerial vehicle (UAV) [Heredia and Ollero, 2010] or a remotely operated vehicle (ROV) [Yuh, 2000] or a mobile robot operating in an urban environment [Lewis, et. al., 2007]. In each case, all information about the system as well as its surroundings is relayed to the controllers via internal sensors that monitor the different system components as well as via external sensors that survey the operating environment.

Consider for example ROVs which are used for a variety of applications like ocean floor surveying and mapping, environmental monitoring and cleanup, construction and inspection of underwater structures like pipelines, etc. [Yuh, 2000]. The vehicles are instrumented with a variety of sensors like video cameras, sonar systems, gyroscopic systems, temperature sensors, etc. intended for different purposes like navigation, mission-specific requirements, diagnostics of the on-board systems, etc. Additional equipment like robotic arms, lights, etc. are also integrated to address various tasks.

These systems may be tethered to a mother ship via umbilical cables which supply power to the different subsystems on the vehicle and also enable bi-directional communication (for sending control signals and receiving video/sensor data) or they may be untethered with onboard power supplied from batteries. The vehicles are remotely controlled by pilots on the ship. Since the ROV operates in unstructured underwater environments, there is always the risk of the umbilical cables being snagged on unknown underwater features and getting damaged or completely severed. [Yuh, 2000] estimates that about ten percent of all ROVs are lost due to such broken tethers[2].

Consider a hypothetical example where the tether is snagged in a location not completely observable by the onboard video camera, thereby impeding the speed with which the ROV moves underwater. Suppose data available from sensors monitoring the speed of the ROV thrusters, due to a bias fault, falsely indicates that they are operating below their intended speed. The pilot may assume that the ROV thruster speed needs to be boosted and in doing so, may risk severing the tether completely and losing the whole vehicle underwater. In such a situation, applying the SPFDI algorithm using a Bayesian network model of the system correlating the thruster speed information with other onboard sensor information like temperature of the thruster pods, current drawn by the thruster motors, the tether orientation available from the video camera, etc., may help the pilot identify the cause of the low speed to be a faulty speed sensor and a snagged tether (for instance, as indicated by an increased current draw and overheating of the thruster motors as indicated by the current and temperature sensor readings)

6.2.4. Performance Level Maximization

The objective in this class of applications is to enable the system to provide the best possible (maximum) performance to address unstructured task requirements or requirements that change while the system is in operation. The systems categorized under this class therefore tend to be highly reconfigurable with additional resources/capabilities

[2] There may be many other factors that may lead to eventual loss of an ROV. For instance, [Griffiths and Brito, 2008] present a Bayesian belief network approach to investigate the risk of such losses occurring for ROVs operating under sea ice environments.

that can be employed to satisfy application demands that lie beyond its nominal design/operating (or duty cycle) specifications when the need arises (without the need for significant retooling or for multiple systems to coexist in tandem to meet specialized needs). The question of what constitutes 'maximum performance' is dictated by the application. For instance, a connotation of this term in the context of specifying performance for a luxury vehicle, could be in characterizing the body roll (which needs to be kept to a minimum) while traversing around sharp turns.

To ensure a desired level of performance, data from the sensors in the application (to monitor the system as well as its operating environment) is used to actively reconfigure and allocate the available system resources to meet the task requirements at any given time. However, if during this procedure, the readings from a particular subset of sensors deviate from their projected values, the decision maker has to determine the dependability of the available values before allocating the available resources. A decision based on faulty sensor data may obstruct the system from performing at its optimum level, even though it has sufficient capability to do so. Or if it is determined that the system capabilities have changed to the extent that it can no longer operate at the desired level of performance, the decision maker may be able to use this information to alter the strategy of reallocating resources to still try and achieve the desired performance goals. The belief values for the different sensors and processes, obtained from the SPFDI algorithm can help address both these issues as described in the following examples.

The emphasis on replacing hydraulic systems and the need for more versatile, light-weight equipment with commonality of platforms to create different types of manned and unmanned ground combat vehicles (GCVs)[3],[4] for different battlefield requirements[5] has led to extensive use of EMAs in such vehicles [Tesar, 2010][Tesar, 2008][Thornhill, et. al., 2003][Doell, 2001]. EMAs are used on advanced GCVs for diverse uses such as drive actuators (wheels/tracks), weapons handling, turret drives, etc.

[3] For greater reconfigurability, rapid technology upgrades, reduced logistics trail, etc.
[4] Ranging from 30lb to 20 tons, and assembled on demand from a set of standard modules [Tesar, 2010]
[5] For instance, vehicles with articulated geometry to address supply, personnel and surveillance needs in rough terrain, armored personnel carriers to protect against explosives, tactical vehicles etc. [Tesar, 2010]

A demanding area of application of EMAs is in active suspensions for these vehicles [Tesar, 2010] where reliable information from sensors is vital for both control as well as condition monitoring.

Most military vehicles are typically overloaded (with armor, equipment, etc.) and cannot adequately perform evasive maneuvers with passive suspension systems. Hence there is a real need for an actively driven suspension system to stabilize the vehicle under severe maneuvers and unfavorable road conditions. Using a suite of sensors to infer the suspension characteristics, vehicle state, and road conditions, active suspensions provide forces that counteract the system dynamics that occur as a result of the transfer of forces from the unsprung mass of the suspension (occurring due to wheel contact with the road surface, sudden maneuvers, road profile etc.) to the sprung mass of the vehicle chassis. By adapting to varying driving conditions (via real-time compensation to the vehicle chassis motions) active suspensions can provide better vehicle handling. They have been an active area of research in applications ranging from passenger cars [Jones, 2005] to future lunar rovers [Bluethmann, et. al., 2010].

Different designs have been explored that utilize linear actuators [Martins, et. al, 2006] [Jones, 2005], rotary actuators in combination with roller screws [Tesar, 2010] [Bluethmann, et. al., 2010], etc. In all such designs a variety of sensors are integrated into the suspension system. Accelerometers are used on the wheels/ chassis (to measure roll, pitch, yaw), absolute linear position sensors are used to gauge the suspension travel; current, voltage and position sensors (rotary/linear) are needed to control the electric prime mover that determines the force generated as well as suspension travel, force sensors to estimate the forces generated from the road profile, etc. [Bluethmann, et. al., 2010][Tesar, 2008] [Jones, 2005][Fischer, et. al. 2003]. The data from all the sensors is sent to the main vehicle controller where algorithms can be implemented to determine the state of the vehicle and road features to determine the force/travel that needs to be generated by each of the suspension actuators.

Consider a situation where a four wheeled Joint Light Tactical Vehicle (JLTV) [Tesar, 2010] is traveling over a rough terrain. The suspension control system has to

accurately determine the rapid motion of the suspension arms on each wheel of the vehicle and the resultant force from the terrain profile acting on the vehicle to produce the necessary compensating forces and keep the vehicle wheels in contact with the road (maintain the chassis as steady as possible). Suppose, the vehicle control system receives readings from the linear position sensors for two of the suspension arms (on the same side of the vehicle) that vary slightly from a steady state value but the sensors for the other two wheel suspensions indicate a large amount of travel. The control system needs to be able to determine if the readings indicate that two wheels of the vehicle are on a relatively smoother section of the terrain and the other two wheels are traversing over a pothole; or if it is indicative of dual failed sensors and if so, on which side of the vehicle.

Using the data from all the sensors involved and relating them using a Bayesian network model for the whole vehicle (consisting of nodes like motor current, motor voltage, linear position for the suspension arms for all the wheels, chassis roll, pitch, yaw orientations, etc.), the SPFDI can help resolve the conflict in the readings. The belief values for the sensors and process (Section 5.4) can help the vehicle control system accurately identify the appropriate course of action.

If the belief values indicate that all the sensors are operating correctly, the vehicle controller can accurately amount of suspension force/travel needed for the wheels in the pothole to keep the chassis level and thus maximize the handling performance of the vehicle. On the other hand, if it is determined that two of the position sensors are faulty[6] then appropriate fault codes to the onboard computers (that may later be used to repair/replace the sensors during scheduled maintenance) while information from the remaining sensors may still be used to generate the appropriate suspension forces to once again provide the best possible vehicle handling performance.

Another relevant example for performance maximization is in the area of manufacturing systems. The demand for a range of high quality affordable products has considerably reduced product life cycle times in many industries (for example, the automobiles sector). With the use of monolithic systems in manufacturing lines, it is not

[6] For example, the two sensors which output a low value, indicating that two wheels of the vehicle are on a smoother section whereas in reality all four wheels of the vehicle may be on extremely rough terrain

easy to respond to sudden changes in product portfolios or market demands. For instance jigs/fixtures designed to aid in the manufacturing of a specific aircraft component may not be usable at all for a different component or a similar component for another aircraft model. This has led to the concept of agile manufacturing.

The defining characteristic is the flexibility of the manufacturing system to reconfigure itself in the shortest possible time (reusing available hardware/software) to respond to unique demands/product needs in a timely manner while simultaneously ensuring an acceptable cost and quality of the end products from such a system [Tesar, 2010][Chen, 2001][Newman, et. al, 2000]. [Gunasekaran and Yusuf, 2002] emphasize that such speed of response to new market opportunities is more relevant to specialty products where quality is also an important consideration in addition to the product cost and throughput, as compared to commodity products which compete solely on the basis of price. Reconfigurable manufacturing cells represent the most common embodiment of such agile manufacturing systems.

The concept of manufacturing cells grew out of metal-machining systems assembled using dedicated precision machines (NC/CNC machines, conveyors, etc.) [Tesar, 2010]. With the advancement in robotics, reliable industrial robots (~100,000 hours of service life [Tesar, 2010]) became the de-facto standard for flexible manufacturing, due to their ability for reprogramming and integration of hardware with relative ease to perform a variety of tasks (like robotic drilling, welding, assembly, etc.) for new product lines.

A typical manufacturing cell includes several such robots working alongside (in parallel) each other to accomplish different steps in the manufacturing process. However with conventional industrial manipulators, the flexibility is still restricted to some extent by the degrees of freedom, hardware/software interfaces of the individual robots (which are usually fixed), lack of absolute accuracy and cell configurations. The use of modular, intelligent EMAs with standard hardware and software interfaces (representing plug and play component technology) can vastly expand this level of flexibility. These EMAs in conjunction with other components like accurate and rigid links, various types of tools,

sensors etc. can be assembled into different configurations (multiple robots) with different degrees of freedom (DOF) on demand [Tesar, 2010].

For instance, a manufacturing cell used for airframe assembly may contain up to 40 DOF composed of two active assembly robots (7 DOF each), two supporting force robots (5 DOF each) and four rigidized fixturing robots (4 DOF each)[Tesar, 2010]. The associated software modules to control these different robots (usually in the form of a set of object oriented libraries like Operational Software Components for Advanced Robotics (OSCAR) developed by [Kapoor and Tesar, 1996]) may also be reprogrammed according to the physical configuration of the manufacturing cell.

A manufacturing cell is a sensor-rich environment since a variety of sensors are integrated into it both for control (for instance, sensors integrated into the joint actuators of the robots that constitute the cell) as well as for safe and reliable operation (for example, sensors that monitor the work area to enable motion planning for multiple robots operating in a common work area, collision avoidance, human safety, etc.). Some of the many types of sensors that may be typically found in an advanced manufacturing cell include optical and ultrasonic sensors, proximity sensors, eddy current and magnetic sensors, contact sensors, machine vision systems, laser-based distance measurement systems, force/torque sensors, rotary and linear displacement sensors, current and voltage sensors, etc. [Tesar, 2010] [Pires and Costa, 2002] [Chen, 2001][Newman, et. al, 2000]. These sensors are used for proximity detection, for precision positioning, robot motion control, obstacle avoidance assembly tasks, etc.

Consider a manufacturing cell with three robots *A*, *B*, *C* where *A* and *B* are used to position the workpiece while *C* performs a drilling operation on it. Assume that only proximity sensors are used to detect the distance between each robot. Suppose the workcell configuration manager software indicates two plausible warnings that a collision has occurred between all the robots. The collision between robots *A* and *C* was signaled based on a zero reading from the proximity sensor monitoring the distance between them but the joint configurations of these robots indicate that a collision should not have occurred. The collision between robots *B* and *C* was signaled based on the joint angles of

the two robots (as measured by the different joint position sensors) but the proximity sensor reading indicates that the two robots are still at a safe distance away from each other. In these unresolved data conflicts, the challenge confronting the supervisor of the workcell is to determine what is actually happening inside the workcell in order to resolve the issue and make a decision on whether or not to proceed with the machining operation in progress.

Using a Bayesian network representation of the complete workcell (with nodes representing the proximity sensors, the joint angles for each robot, the current and voltage for each joint actuator of each robot, drilling force, drilling speed, etc.), the SPFDI algorithm can determine belief values for the different sensors and processes in the workcell. These values may indicate that the collision between robots *A* and *C* was a false alarm caused by a faulty proximity sensor and the collision between robots *B* and *C* was a real collision caused by dual faults in the proximity sensor as well as a process fault that caused an error in the joint motion of one of the robots (for instance, due to a joint actuator failure resulting from overheated motor windings). With this knowledge, the workcell supervisor may be able to reprogram the control software to take into account these faulty sensors and processes and determine the best possible method to reconfigure the remaining available resources to accomplish the drilling operation.

6.3. HUMAN SAFETY AT STAKE

Conventional braking systems used in most automobiles have been comprised of mechanical friction brakes actuated by a system of links and hydraulic actuators that amplify the input brake pedal force applied by the driver. With the advent of anti-lock braking systems, in addition to the decision by the driver as to when and how to apply the brakes, the brakes are also partly computer controlled using a suite of sensors and an electronic brake control unit (EBCU) to prevent the locking-up or slipping of wheels during emergency maneuvers. These advances in braking systems, the developing trend towards regenerative braking in hybrid and electric vehicles, the growing emphasis on all-electric modular automobiles [Tesar, 2010] and the various issues associated with

hydraulic systems in general[7] have led to electromechanical braking systems (referred to as 'brake-by-wire' systems) gaining in popularity [Winkler, et. al, 2005]

A typical brake-by-wire system consists of individual units attached to the different wheels. All the units are controlled and coordinated by a central EBCU, in order to provide self-sufficient brake control to each wheel of the vehicle (Figure 6-1). Each braking unit includes an EMA with an electric prime mover (typically a three phase brushless DC motor) coupled to a reduction geartrain. This arrangement is then connected to a ball screw to convert the rotary motion into linear motion that can then generate required actuation/braking forces on the brake calipers [Saric, et. al, 2008]. Each of the units can act independently and are instrumented with a suite of multiple sensors for feedback control of the rotary prime mover, the ball screw, measuring the clamping force, etc. The multiplicity of sensors used also increases the inherent diagnostic capabilities afforded by the system.

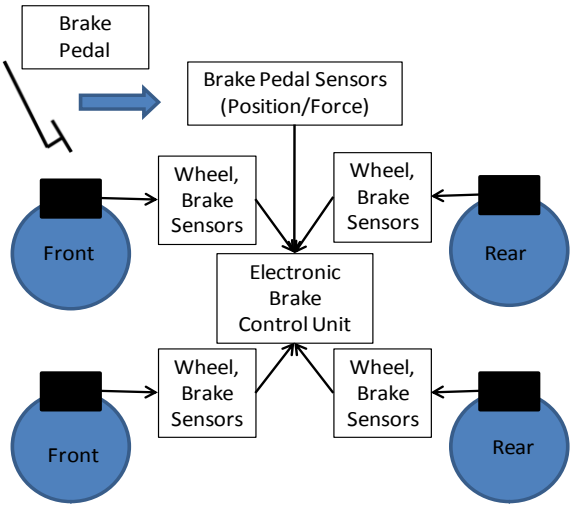


Figure 6-1: Schematic of a Brake-by-Wire System

The prime advantage of brake-by-wire systems is their quicker responsiveness compared to traditional hydraulic systems. With independent braking units, there is potential to implement sophisticated active braking programs to improve the vehicle handling and performance when it is traveling over different types of terrains (for instance, improved fuel economy in city driving, better handling on icy roads, etc.). The fewer parts required for such systems, also results in an overall reduced life cycle cost. Then, significant maintenance costs are restricted to only the brake pads and discs.

[7] Higher weight for hydraulic cylinders adding to the overall weight at each wheel which affects the overall vehicle weight as well as its fuel economy and handling, the need for complicated hydraulic lines to run through the vehicle, regular replenishment of hydraulic/braking fluids, maintenance, etc

Each time the driver applies a force on the brake lever, the EBCU requires feedback from the pedal to determine the user intent (how much force needs to be applied based on the brake lever position and the urgency based on how fast the brake lever has moved). Feedback is also required from the individual brake units to determine the amount of travel needed for each of the calipers to apply the desired braking force and distribution of the braking forces on each individual wheel to maintain control of the vehicle and avoid skidding, slipping, etc. The action of the driver's foot on the brake pedal is used to modulate the control signals that move the EMAs at each wheel to actuate the friction surfaces. Position and force sensors are typically integrated with the pedals in brake-by-wire systems to gauge the user intent. To determine the magnitude and distribution of forces to be applied at each wheel, wheel speed sensors, motor current and position sensors, and brake caliper force sensors are typically used [Doriben and Durkopp, 2003][Pisu, et. al, 2003][Fleming, 2001].

As with any system where human safety is paramount, there is a need to guarantee fault-free operation for all brake-by-wire systems or identify a fault as soon as it occurs so that remedial action is possible. For instance, [Pisu, et. al., 2003] investigate a model-based fault detection and isolation scheme to detect and identify faults in the caliper force and current sensors using a residual generation methodology. With different types of sensors available in a brake by wire system, it is possible to reduce the dependence on additional hardware and rely on the information redundancy that is made possible by these sensors to provide this assurance. For instance, [Saric, et. al., 2008] demonstrate the use of position and motor current sensors instead to determine the brake clamping force, which can prove valuable in case of a failure of the force sensor or if a decision is made to leave out the sensor altogether from the brake module (primarily for cost reasons). By employing similar synergy between the sensors the SPFDI algorithm can not only identify faulty sensors but also potential faults in one or more of the other components of the braking system as discussed in the example below.

Consider a single brake module. Figure 6-2 shows the Bayesian network representation for the system that can be developed by the domain expert based on causal

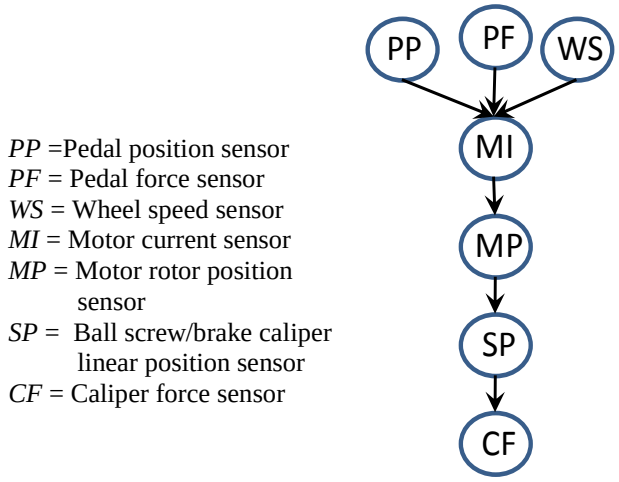


Figure 6-2: Representative Bayesian Network for a Brake by Wire System

reasoning as explained in Chapters 2, 3 and 4. Using feedback information regarding the brake pedal position (*PP*), the pedal force with which it is moved (*PF*), and the wheel speed (*WS*), obtained from the respective sensors, the EBCU modulates the current applied to the EMA prime mover, measured by the current sensor (*MI*). This, in turn determines the rotation of the motor as determined by the rotor position sensor (*MP*). The motor rotation controls the linear motion of the ball screw and in turn the position of the caliper, indicated by the linear motion position sensor (*SP*). Finally, the braking force as determined by the position of the ball screw is measured using the caliper force sensor (*CF*).

Table 6-2: Example of an Instantiation Table (for Figure 6-2)

PP	**PF**	**WS**	**MI**	**MP**	**SP**	**CF**	**PP->MI**	**PF->MI**	**WS->MI**	**MI->MP**	**MP->SP**	**SP->CF**
NI	*NI*	*NI*	*NoI*				*PoI*	*PoI*	*PoI*			
NI	*NI*	*NI*		*NoI*			*PoI*	*PoI*	*PoI*	*PoI*		
NI	*NI*	*NI*			*NoI*		*PoI*	*PoI*	*PoI*	*PoI*	*PoI*	
NI	*NI*	*NI*				*NoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*
			NI	*NoI*						*PoI*		
			NI		*NoI*					*PoI*	*PoI*	
			NI			*NoI*				*PoI*	*PoI*	*PoI*
				NI	*NoI*						*PoI*	
				NI		*NoI*					*PoI*	*PoI*
					NI	*NoI*						*PoI*

Table 6-2 shows an example of an instantiation table that may be generated for this network (Section 5.4). If during a braking operation, all the sensors and processes are operating correctly, the W_S and W_P belief values would attain a value of 1 as shown in Table 6-3 (starting with values of 0.5, as explained in Chapter 5, Section 5.4.2).

Table 6-3: W_S and W_P Values with No Sensor/Process Faults (for Figure 6-2)

PP	PF	WS	MI	MP	SP	CF	PP->MI	PF->MI	WS->MI	MI->MP	MP->SP	SP->CF
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

In this situation, since all the critical components have been determined to be operating normally, the EBCU can decide on the appropriate distribution of the braking forces between the different braking units and indicate, for instance using a green lamp on the car dashboard, that the system is working normally. This information can then be utilized by the driver who can then factor in the assurance that the braking system is operating normally, while deciding on how to steer or accelerate the vehicle. Suppose, the W_S and W_P belief values obtained under the same situation as above are now as follows:

Table 6-4: W_S and W_P Values with a Sensor Fault (for Figure 6-2)

PP	PF	WS	MI	MP	SP	CF	PP->MI	PF->MI	WS->MI	MI->MP	MP->SP	SP->CF
0.75	0.75	0.75	0.75	0.75	**0.000**	0.75	0.75	0.75	0.75	0.6666	0.75	0.75

A W_S value of 0.000 for the linear motion sensor *SP* indicates a possible fault in it (confirmed after multiple iterations of the SPFDI algorithm). However, since all the other W_S and W_P belief values are higher, the EBCU may use this information to indicate that the brake system is functioning normally overall but also indicate via a separate warning light on the car dashboard (or a suitable maintenance diagnostic code) that the caliper force sensor requires maintenance. This information is again useful to the driver since it provides advance warning that a particular component of the system needs repair though it does not critically affect the system performance immediately. In such a case, as a back-up provision, the value of the caliper force may be inferred by the EBCU using the values from the other sensors (for instance, *MP* or *CF* which have higher belief values, and the processes *MP->SP* and *SP->CF* are also deemed to be working correctly as indicated by their belief values) as evidence in an appropriate inferencing algorithm. Now consider a situation where the driver presses on the brake pedal but notices that the braking action is not adequate to bring the vehicle to halt quickly. In such a situation, suppose the W_S and W_P belief values obtained from the SPFDI algorithm are as follows:

Table 6-5: W_S and W_P Values with a Process Fault (for Figure 6-2)

PP	PF	WS	MI	MP	SP	CF	PP->MI	PF->MI	WS->MI	MI->MP	MP->SP	SP->CF
0.500	0.500	0.500	0.500	0.500	0.250	0.25	0.500	0.500	0.500	0.333	**0.000**	0.25

In this situation, it can be seen that that the W_P value for the *MP->SP* process attains a value of zero whereas the belief values for all the other sensors and processes in the system are higher. This implies that for a given rotation of the motor the brake pad caliper has not advanced the distance it was supposed to in order to generate the required braking force. This could be indicative of excessive wear of the brake pad friction material which may be causing it to lose contact with the brake disc, leading to the potential loss of braking ability. The vehicle driver may be able to again access this valuable information from the EBCU to diagnose the root cause of the problem.

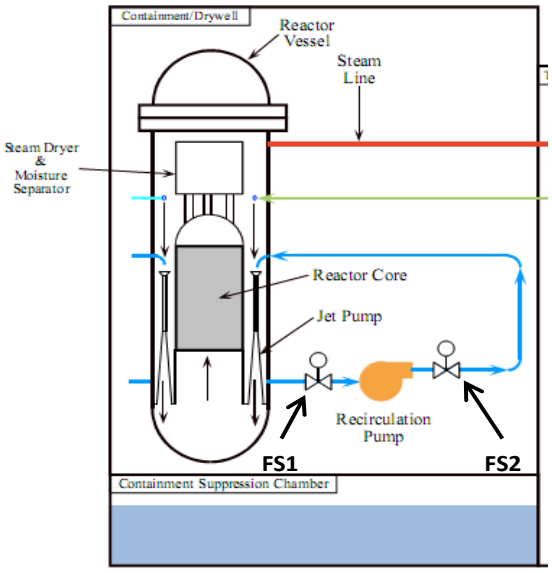
6.4. SYSTEM AVAILABILITY

The following section presents an in-depth scenario analysis for system availability. Consider a nuclear power plant, that is powered down for maintenance purposes and a decision has to be made regarding whether or not to ramp it up to its optimum capacity again. But before doing so, it must be ensured that all the parts of the plant are operating as intended by running a series of prescribed system checks.

The reactors in a typical nuclear power plant generate heat (by a fission reaction) that is removed by circulating a coolant around it; the process is also used to convert the generated heat into electrical power. In the simplest form, a conventional coolant circulation system consists of a pump and a series of flow regulating valves. Any reduction or a complete loss of coolant flow may be due to an actual fault in the pumps used in the coolant circulation system (for instance, if the pump is unable to operate at its rated volumetric output due to a damaged bearing or lubrication system [Duran, 2003]).

In other cases, the reduction may be due to a control system action triggered by a false alarm from the flow sensors that are used to monitor and determine the valve positions to control the flow (for instance, fouling of the sensing element may occur due to build-up of particulates dissolved in the water [Gross et. al, 1997]). Such situations

may result in overheating of the core rods. This can not only damage the core of the reactor but is also considered one of the potential modes of complete failure for a nuclear reactor. Hence, it is necessary to ensure proper functioning of the coolant circulation system before the plant generation capacity can be ramped up.



[USNRC, 2010]

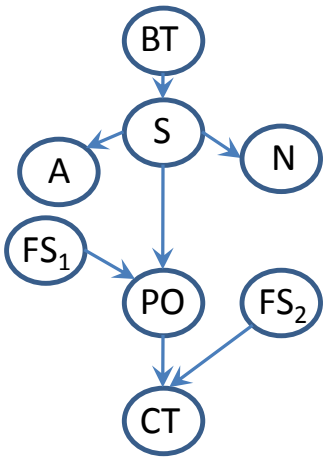
Figure 6-3: Boiling Water Reactor Schematic

Ensuring a steady flow of coolant is not only crucial for safety but also for efficient operation as the coolant flow rate is used in thermal power calculations that determine the per unit cost of energy [Gross et. al, 1997][8]. To avoid such occurrences, every power plant is outfitted with a number of emergency systems, both passive (inherently built into the reactor design) as well as multiple active safety systems (representing multiple levels of redundancy).

However, in addition to adding to the overall cost, the back-up systems are themselves subject to some of the same issues facing the primary coolant circulation system. With any unanticipated deviations in the readings from the monitoring sensors, it is necessary to determine whether there is an actual issue with one of the circulation system components or if they are the result of fouled sensors. By utilizing the different sensors already incorporated into the primary and the back-up circulation systems it may be possible to avoid false alarms and enable condition based maintenance of any degrading components and thereby avoid any unforeseen system shutdowns. This approach is also emphasized by [Kang and Golay, 1999] who stress the need for an advisory system geared towards ensuring operational availability, in addition to safety.

[8] [Gross et. al 1997] estimate that 2% power derating due to reduction in flow rate may cost as much as $7.3 $M per year, in lost revenue for an typical power plant

Consider a simplified situation where the flow to the reactor core and hence the core temperature is influenced only by the operation of the pump and the flow valves (Figure 6-3). It is assumed here that the performance of the pump in terms of its operational speed is determined solely based on the condition of its bearings. The volumetric output from the pump is determined by its operating speed and the amount of coolant flowing into the pump, controlled by a valve based on signals from a flow sensor. The output of the pump is similarly controlled by another valve that uses a second flow sensor. The total amount of coolant that flows around the reactor core, and hence its resultant temperature is then determined by the pump output. The Bayesian network for this system is shown in Figure 6-4[9].



B T = Bearing temperature
A = Accelerometer (vibration)
N = Acoustic noise
S = Pump operating speed
PO = Volumetric pump output
CT = Reactor core temperature
FS1, *FS2* = Flow sensors

Figure 6-4: Representative Bayesian Network for a Coolant Circulation System

Suppose that a temperature sensor (*BT*) is used to evaluate the lubricant viscosity in order to assess the condition of the bearings in the pump[10]. In addition, a speed sensor (*S*) and an acoustic noise sensor (*N*) are also used to monitor the condition of the pump. The flow to the pump and the pump output are controlled by two valves (*FS1* and *FS2*). The pump output is measured by the sensor *PO*. The core temperature is monitored by another temperature sensor (*CT*).

[9] Similar approaches, some with more comprehensive Bayesian networks spanning more subsystems in a power plant have also been proposed by [Heo, et. al., 2005], [Duran, 2003], [Kang and Golay, 1999], etc. However, in each case the goal was to create a knowledge base for fault diagnosis purposes rather than for fault detection and isolation (i.e. there was no distinction between sensor faults and system faults)

[10] Any issue in the bearing rolling elements due to pitting or misalignment creates additional friction which leads to higher temperatures. This changes the lubricant quality (reduced viscosity) and makes film formation more difficult. This, in turn, sets up a cycle of higher friction and further temperature rise. The temperature rise also affects bearing clearances [Kang and Tesar, 2004]

Table 6-6 shows one possible instantiation table that may be generated for this network. If at the end of a series of fault detection and isolation cycles that are executed during the start-up process, all the sensors and processes represented in the validation table above are operating normally, then the W_S and W_P belief values would attain a value of 1 (Table 6-7) (Section 5.4.2.3).

Table 6-6: Example of an Instantiation Table (for Figure 6-3)

BT	S	A	N	FS1	FS2	PO	CT	BT->S	S->A	S->PO	S->N	FS1->PO	PO->CT	FS2->CT
NI	*NoI*							*PoI*						
NI		*NoI*						*PoI*	*PoI*					
NI			*NoI*					*PoI*			*PoI*			
NI						*NoI*		*PoI*		*PoI*				
NI							*NoI*	*PoI*		*PoI*			*PoI*	
	NI	*NoI*							*PoI*					
	NI		*NoI*								*PoI*			
	NI					*NoI*				*PoI*				
	NI						*NoI*			*PoI*			*PoI*	
				NI		*NoI*						*PoI*		
				NI	*NI*		*NoI*					*PoI*	*PoI*	*PoI*
					NI	*NI*	*NoI*						*PoI*	*PoI*

Table 6-7: W_S and W_P Values with No Sensor/Process Faults (for Figure 6-3)

BT	S	A	N	FS1	FS2	PO	CT
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

BT->S	S->A	S->PO	S->N	FS1->PO	PO->CT	FS2->CT
1.000	1.000	1.000	1.000	1.000	1.000	1.000

In this situation, the decision maker has complete confidence to proceed with the remainder of the start-up procedure for the plant since all the critical components have been determined to be operating normally. Now consider a situation where the W_S and W_P belief values for the sensors and processes obtained are as follows:

Table 6-8: W_S and W_P Values with a Sensor Fault (for Figure 6-3)

BT	S	A	N	FS1	FS2	PO	CT
0.800	**0.000**	0.500	0.500	1.000	1.000	0.75	0.75

BT->S	S->A	S->PO	S->N	FS1->PO	PO->CT	FS2->CT
0.800	0.500	0.500	0.500	1.000	0.75	1.000

These values indicate a definite fault in the pump speed sensor (as indicated by W_S =0). But since the other W_S and W_P belief values are higher, the decision maker may now use the confirmed knowledge that the speed sensor is indeed faulty and may decide to replace/fix it (if it can be done relatively quickly) or may choose to continue operating with the faulty sensor if it is not deemed to be critical at that moment, before proceeding with the remainder of the power plant start up procedure. If the decision maker decides to take the faulty sensor out of service, then the pump speed may still be estimated using the other sensors (*BT*, *A*, *N* as long as they are determined to be non-faulty) by using their values as evidence in inferencing algorithms[11].

Table 6-9: W_S and W_P Values with a Process Fault (for Figure 6-3)

BT	**S**	**A**	**N**	**FS1**	**FS2**	**PO**	**CT**
0.600	0.600	1.000	1.000	0.500	1.000	0.500	0.500

BT->S	**S->A**	**S->PO**	**S->N**	**FS1->PO**	**PO->CT**	**FS2->CT**
0.600	1.000	**0.000**	1.000	0.500	0.500	1.000

Now consider the situation where the operator notices a drop in the coolant flow and a slightly higher core temperature than expected. The challenge then would be to determine if the corresponding *CT* and *PO* sensors are faulty or if there is some other problem that is causing the variation in the readings. Suppose the W_S and W_P belief values for the sensors and processes obtained are as shown in Table 6-9 .A value of zero for the *S->PO* process in the above case indicates that for a given operating speed, the pump is unable to produce the desired output (for instance, due to an internal blockage) but based on the valve positions *FS1* and *FS2* and the actually available output *PO*, the core temperature *CT* is what it is supposed to be based on the relationships encoded in the network. In this set of results, however, there is a clear indication to the system operator to immediately halt the operation and rectify any issues with the pump before proceeding, in order to avoid more serious consequences once the plant is operating at its full capacity.

[11]As explained in Chapter 4 (Section 4.3.4), the uncertainty in the inferred values of *S* would depend on the quality of the test data/performance maps that are used to create the conditional probability tables representing the relationships *BT->S*, *S->A* and *S->N*

6.5. HIGH COST OF SYSTEM FAILURE

Energy production from renewable sources has gained greater traction over the past decades, to eliminate emissions that arise from other methods using hydrocarbons. Wind energy has been proven to be one of the more successful and commercially viable alternatives. With longer, lighter, efficient and aerodynamic blades, better structural designs, etc. the output from wind turbines has been increasing constantly. Large turbines (producing energy of the order of MW) are usually installed in large groups in offshore locations or other remote locations to harness the wind. These wind farms represent a high value investment (with the cost for each unit running into hundreds of thousands of dollars). Figure 6-5 shows the schematic for a typical horizontal axis turbine. The pitch and yaw of the rotor blades are continuously tuned by computers to best align them with the wind direction and control the power generated [White et. al., 2009]. The blades are attached to a hub on the input shaft of the gearbox which amplifies the blade rotation speed to the speed required by the generator (from ~25-60 rpm/high torque to 1000-1800 rpm/low torque) [Rensselar, 2010]).

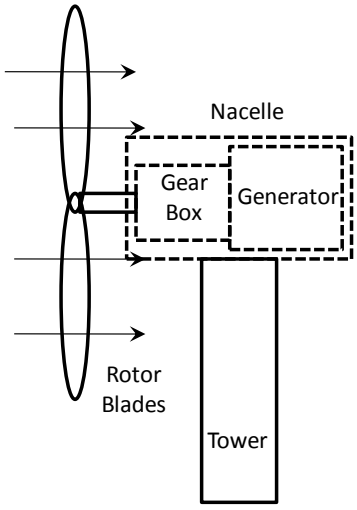


Figure 6-5: Schematic of a Wind Turbine

The different components of a turbine i.e. rotor blades, gearbox, generator, bearings, nacelle, the support column, etc. are all exposed to extreme environmental conditions. They are subject to severe temperature gradient changes, inclement weather, wind loads, icing, etc. Such exposure affects the life of all these components. Hence, despite all the advancements in design and manufacturing, reliability is still a critical issue for wind turbines, especially for all the rotational components of the system. These components operate under drastically varying dynamic loads that may result in different modes of failure in each of them [Hameed, et. al., 2009]. Minor faults (for instance, incipient wear in gear teeth, overheating of generator windings, etc.) that remain undetected may lead to faults in other sub-systems and eventually cause a catastrophic

failure of the whole turbine. [Rolfes, et. al., 2007] estimate that wind turbines may sustain such serious unforeseen damage up to five times a year.

The relative inaccessibility of these systems implies that regular inspection by experts to detect any faults or frequent maintenance is tricky to implement. With an unforeseen breakdown of any turbine component, the cost of repair may not only be prohibitively expensive (in terms of actual cost of parts as well as logistics required to carry out the repairs), but can also cause other significant financial losses in terms of lost capacity for power generation during the downtime. Hence there is a need for comprehensive remote condition monitoring that can detect the first signs of any faults that may develop in the turbine components. [Hameed, et. al., 2009] emphasize that proactive condition-based maintenance (CBM) can be more beneficial than any corrective maintenance. With CBM, the replacement of parts which are still functional may be avoided during scheduled maintenance and the time period between successive inspections may also be increased; thereby minimizing overall life-cycle costs[12].

For both effective control and CBM, a suite of sensors are needed as described by [Hameed, et. al., 2009], [Odgaard, et. al., 2009], [Wei, et. al., 2009], [Rolfes, et. al., 2007], etc. The pitch angle of the rotor blades and the position/speed of the gear/generator shafts, etc. are measured using absolute position sensors. The vibrations induced in the nacelle by rotary components may be measured with accelerometers. Acoustic noise sensors may be used to measure windage noise generated by the same components. Temperature sensors may be used to assess lubricant quality (ensure that the existing temperature allows the sufficient lubricant viscosity), monitoring the generator windings (for hot spots), etc. The wind speed, load acting rotor blades, their resulting deformation, etc. may be monitored using fiber optic sensors, acoustic emission sensors, strain gages, accelerometers, inertial/ angular rate sensors etc. Current and voltage

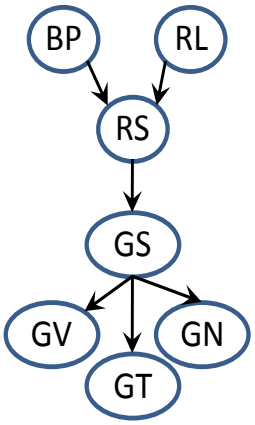
[12][Hameed, et. al., 2009] provide an inclusive review of a variety of methods (observer-based, expert systems, empirical methods, etc.) and their application to various parts of a turbine (rotor blades, gearbox, bearings, lubricating system, generator, etc.) for both condition monitoring and fault detection. [Rothenhagen and Fuchs, 2008] present a model-based approach to fault tolerant control of generators under sensor failures (using observers for current/ voltage). [Wei, et. al. 2009] also present an observer based scheme for detection of sensor faults for blade root torque sensors.

sensors may be used to monitor the electrical output from the turbine and the power electronics associated with the unit. A torque sensor may be used to control the generator/converter torque.

However, one or more faulty sensors may result in a false alarm from the CBM algorithms; either calling for expensive and unscheduled maintenance or failing to detect the occurrence of an actual fault in a turbine component. If undue variations are observed in any of the sensor readings, and if it can be attributed to the occurrence of a specific sensor or process (system) fault with some certainty, the supervisor can decide if a maintenance operation is immediately needed to avoid a turbine failure or if the available resources in the sensor-rich environment can be reconfigured for continued operation. This can be achieved through the SPFDI algorithm as discussed below.

The cyclic wind loads on the rotor blades induces vibrations and oscillations in all the components. The resultant cyclic stresses may not only cause cracks to develop in the rotor hub/blades due to fatigue [White, et. al., 2009] but can also affect the gearbox. A large portion of wind turbines' downtime can be attributed to failed gearboxes [Rennselar, 2010] which represent a significant portion of the cost of parts as well as the operating cost of a turbine. [Musial, et. al, 2007] state that a bulk of wind turbine gearbox failures start off as bearing failures rather than the gear teeth failures. As the blades rotate and pivot, the resulting gyroscopic precession produces tremendous forces that can cause spalling of the rotor shaft bearings and result in cracked bearing races [Drewry and Georgiou, 2007]. The debris from the cracked races, in turn, may cause abrasion/cracks of other components like the gear teeth, shaft, etc., resulting in eventual failure of the whole gearbox. In addition, inadequate lubricant film formation (due to changes in viscosity characteristics at elevated temperatures or squeezing out of the lubricant due excessive bearing loads caused by strong winds) may also contribute to bearing damage.

Figure 6-6 shows an example of a Bayesian network (based on causal relationships) that can be developed for a wind turbine gearbox by the domain expert (the variable represented by each node in the network is measurable using an appropriate sensor). The gearbox input speed (*RS*) is determined by the blade pitch angle (*BP*) and



BP = Blade pitch angle
RL = Wind load acting on the rotor
RS = Rotor speed = Gearbox input speed
GS = Gearbox output speed = Generator operating speed
GV = Gearbox vibration amplitude
GT = Gearbox temperature
GN = Acoustic noise generated by the gearbox

Figure 6-6: Representative Bayesian Network for a Wind Turbine Gearbox

the wind load on the rotor (*RL*). Taking into account the gear ratio, the gearbox output speed (*GS*) can be calculated. Assume that the vibration (*GV*), temperature (*GT*) and acoustic noise (*GN*) characteristics of the gearbox are monitored for CBM purposes. Table 6-10 shows a possible instantiation table that may be generated for this network (Section 5.4). If during regular operation of the turbine, all the sensors and processes represented in the network are operating correctly, the respective W_S and W_P belief values would attain a value of 1 as shown in Table 6-11 (starting with initial values of 0.5, (Section 5.4.2)). The belief values in Table 6-11 indicate that all the components of the gearbox, including the sensors are operating correctly. The signs of bearing damage are typically manifested in the form of excessive ripples in the gearbox output/generator operating speed[13].

Table 6-10: Example of an Instantiation Table (for Figure 6-6)

BP	**RL**	**RS**	**GS**	**GV**	**GT**	**GN**	**BP->RS**	**RL->RS**	**RS->GS**	**GS->GV**	**GS->GT**	**GS->GN**
NI	*NI*	*NoI*					*PoI*	*PoI*				
NI	*NI*		*NoI*				*PoI*	*PoI*	*PoI*			
NI	*NI*			*NoI*			*PoI*	*PoI*	*PoI*	*PoI*		
NI	*NI*				*NoI*		*PoI*	*PoI*	*PoI*		*PoI*	
NI	*NI*					*NoI*	*PoI*	*PoI*	*PoI*			*PoI*
		NI	*NoI*						*PoI*			
		NI		*NoI*					*PoI*	*PoI*		
		NI			*NoI*				*PoI*		*PoI*	
		NI				*NoI*			*PoI*			*PoI*
			NI	*NoI*						*PoI*		
			NI		*NoI*						*PoI*	
			NI			*NoI*						*PoI*

[13] As well as elevated levels of vibration, acoustic noise, and temperature

Table 6-11: W_S and W_P Values with No Sensor/Process Faults (for Figure 6-6)

BP	RL	RS	GS	GV	GT	GN	BP->RS	RL->RS	RS->GS	GS->GV	GS->GT	GS->GN
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

If it is observed that the gear speed readings vary a lot from a steady speed required for efficient generator operation, the decision maker needs to decide if the variation is due to a faulty gear speed sensor or if it is due to an incipient fault in one of the gearbox bearings. Suppose the W_S and W_P belief values obtained from the SPFDI algorithm are as follows:

Table 6-12: W_S and W_P Values with a Sensor Fault (for Figure 6-6)

BP	RL	RS	GS	GV	GT	GN	BP->RS	RL->RS	RS->GS	GS->GV	GS->GT	GS->GN
0.800	0.800	0.800	**0.000**	0.667	0.667	0.667	0.8	0.8	0.75	0.667	0.667	0.667

If the values shown above are obtained after multiple iterations of the SPFDI algorithm, then the decision maker can be assured of the fact that the ripples in the gearbox output speed are a result of a faulty speed sensor (as indicated by W_{GS} = 0). In this case, the decision maker may decide to hold off on repairing or replacing the sensor until the next scheduled maintenance operation if it is known (based on experience or other knowledge) that the faulty sensor does not hamper the operation of the turbine otherwise. The higher belief values for all the other sensors and processes in the system, also indicates that any of the other sensors may be used to infer the gearbox output speed if necessary.

Now, consider a situation, where under a known set of operating conditions[14], the turbine supervisor not only notices ripples in the gearbox output speed but also unusual readings from the accelerometer, temperature and acoustic noise sensors. Suppose, the W_S and W_P belief values obtained in this case are as shown in Table 6-13. It can seen from the above values that the W_P value for the *RS->GS* process attains a value of zero thereby indicating a potential fault in the process of speed amplification in the gearbox (which is most likely due to an incipient bearing fault as described earlier).

[14] Where wind load, indicated by the corresponding sensor, is as per the value that could be expected under the prevailing weather conditions and the blade pitch angle is set by the control system to maximize the turbine efficiency

Table 6-13: W_S and W_P Values with a Process Fault (for Figure 6-6)

BP	RL	RS	GS	GV	GT	GN	BP->RS	RL->RS	RS->GS	GS->GV	GS->GT	GS->GN
0.200	0.200	0.200	0.600	0.400	0.400	0.400	0.200	0.200	**0.000**	0.400	0.400	0.400

Subsequent iterations of the SPFDI algorithm can help confirm this fact and also provide assurance that the deviations in the values of multiple sensors are not indicative of multiple sensor faults (as indicated by their higher belief values). In this case, based on other factors like the pre-determined safety thresholds on actual amplitude of vibrations, acoustic noise temperature rise, etc. the decision-maker may decide to temporarily shut down the turbine to carry out an emergency inspection and perform the necessary repairs to avoid the risk of losing the whole turbine.

6.6. PERFORMANCE MAXIMIZATION

Soldiers are often engaged in harsh and long duration combat missions or rigorous non-combat training conditions where they are required to sustain high levels of performance[15] under widely varying conditions. The demands imposed on them in such situations often approach or exceed the limits of their physical and mental capabilities. The availability of sophisticated equipment notwithstanding, the success or failure of any mission is largely dictated by the soldiers' ability to carry out the required duties. The goal is thus to maximize the performance of each soldier as a fundamental element critical to the effectiveness of the larger platoon as well as to the success of the missions in which they are deployed.

The operational performance of warfighters in both combat and training conditions is influenced by a range of factors including (but not limited to) their prior conditioning, inherent cognitive and physical abilities, terrain, environmental factors (heat, rain/snow, altitude, etc.), command tactics, resting condition and nutrition, etc.[Valk and Veenstra, 2009]. Of these, the prime factors of interest for commanders are the physiological and cognitive capabilities of their troops.

[15] Retain continuous situational awareness and vigilance, overcome sleep/nutrition deprivation, perform physically challenging tasks in extreme heat or cold conditions, make quick decisions under threat, carry out tasks requiring exceptional dexterity/motor skills, maintain clarity of communications, etc.

Each soldier represents a highly trained, high-valued asset with an array of physical and cognitive skills acquired through years of training (representing the resources at the disposal of that individual). The commanders need to be able to accurately estimate the exact capabilities for each soldier in order to judge their ability to perform optimally under stressful combat or training conditions. If the commander can estimate accurately that specific abilities of a soldier are not up to the mark to achieve the desired proficiency for some task, then the commander can suggest a suitable 'reconfiguration' of the soldier's resources to maximize the soldier's performance for that task. Alternatively, such evaluation may also be used by commanders to decide how the soldiers in a platoon may be allocated to different combat functions based on their extant capability or to decide if certain individuals could be pushed harder during training.

To assess a soldier's physiological or cognitive condition prior to a mission or training exercise, commanders have typically relied on their experience and available physical cues (obvious signs of injury, fatigue, dehydration, sleep deprivation, analysis of gait, slurred speech, dilation of pupils, etc.). There are many benefits that both the commanders and the soldiers themselves may be able to derive from these assessments (since they provide explicit and implicit signs of any incipient deterioration in the soldiers' performance) such as reduced risk of overuse injury, early detection of stress, exhaustion, detecting abnormal changes in body core temperature or glycemic levels, etc. [Friedl, 2007].

However, with subjective evaluations, despite a commander's extensive experience there is always the potential for a wrong assessment (or purposeful misrepresentation) to occur, especially if some of the available physical cues are conflicting[16]. In some cases, by the time any visible symptoms for an issue that may affect the soldier's performance (say, dehydration or fatigue) are observed, it may already be too late and medical help may be required. This may not only put an individual soldier at risk but also compromise the effectiveness of the platoon. Also, once a mission or training exercise is underway, there is often little opportunity to reassess in detail the

[16] It is entirely possible for the soldier to be unaware of an incipient critical condition, or if the soldier wishes not to admit it or is not enabled to take timely self-corrective action

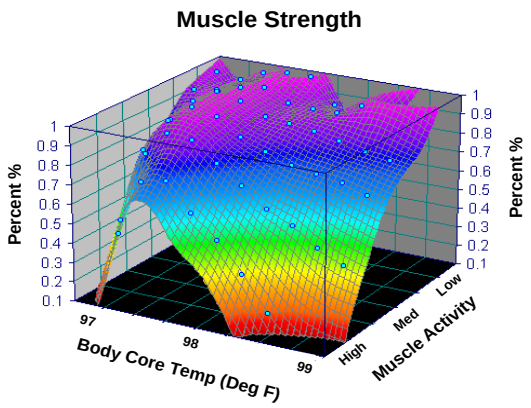
soldiers' condition and evaluate whether their performance capability has deteriorated in any manner.

In order to address these shortcomings, there has been an increasing emphasis on real-time monitoring of soldiers' cognitive and physical abilities using an array of unobtrusive and noninvasive sensors [Tesar, 2010][Freund, B., 2008][Friedl, 2007]. Through continuous real-time monitoring of various physiological parameters, valuable data can be obtained as to how each individual responds during stressful and long duration operations. Such data may help the commander determine if a soldier is suffering from heat exhaustion or if a soldier is prone to accidents or errors due to sleep deprivation, etc. Since medical resources are limited in combat situations, an added benefit offered by such an approach is that it also may allow medics to monitor the vital signs of soldiers remotely.

The data may be integrated with soldier performance prediction models/maps [McFarland and Tesar, 2010] to aid the commanders' strategy and decision-making on how to best utilize and maximize an individual soldier's performance capabilities, or how quickly any performance degradation may be detected for a particular soldier in future missions (Figure 6-7). It may also assist in other logistical aspects of mission-planning or training such as determining the needs for each soldier (nutrition, hydration, sleep, etc.) to enable them to function effectively in different environments or customize the training for individuals to address specific shortcomings in their skills and physical ability.

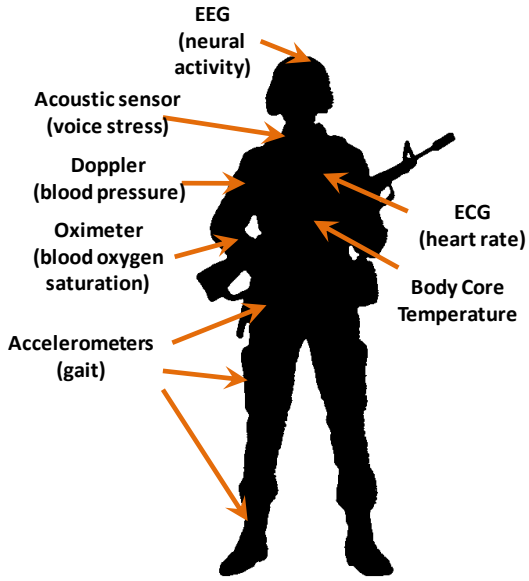
The concept of using wearable sensors has been implemented in monitoring of vital signs of patients in hospitals (to provide warnings when the physiological signals vary significantly beyond the normal acceptable range) [Jafari., et. al., 2005], use of heart-rate monitors, pedometers etc. by athletes during training, etc. Equipping a soldier with multiple sensors and continuous monitoring to prevent a soldier 'system' degradation (or complete collapse in the extreme case) is similar to CBM strategies adopted for military vehicles to allow them to function at optimum performance levels for long durations with nominal maintenance [Friedl and Allan, 2004].

Sensors for monitoring various physiological parameters may be integrated at



[Tesar, 2010 (3)]

Figure 6-7: Conceptual Performance Map for Operational Soldier Performance /Resilience



[Background Image Source: http://www.au.af.mil/au/awc/cliparmy/1-1b.gif]

Figure 6-8: Warfighter Monitoring Using Multiple Sensors

specific locations into a garment worn by the soldier (helmet-mounted, wrist-worn, and harness configurations are also possible), in addition to miniaturized batteries, data acquisition and signal processing electronics, radio transceivers, etc. Sensor data may be transmitted wirelessly either to field commanders or to commanders located remotely, who can monitor these parameters continuously and correlate them to assess the overall health and performance capability of the soldier.

A number of such sensor suites have been proposed/ developed by R&D establishments as well as commercial suppliers. These include systems as suggested in [Tesar, 2010 (3)], the Warfighter Physiological Status Monitor (WPSM) [Burns, P.G., 2009][Friedl and Allan, 2004], BioHarness™[Zephyr Technology, 2010], TrainTrak™ [Qinetiq NA, 2010], LifeGuard [Mundt, et. al. 2005], MIThril[Sung, et.al., 2004], the Georgia Tech Smart Shirt [Gopalsamy, et.al.,1999], etc.

The typical parameters monitored using such systems include the heart rate, blood pressure, galvanic skin response, body core temperature, skin temperature, body position/posture, blink rate, etc. The

sensors that are typically integrated into such systems include heart rate monitors (Electrocardiogram (ECG)), blood pressure, temperature sensors (in the form of pills swallowed by soldiers), accelerometers, actigraph sensors (for sleep monitoring), Electromyogram (EMG, for muscle activity), Electroencephalogram (EEG, for neural activity), acoustic sensors for voice stress analysis, pulse oximeters (for blood oxygen saturation levels), etc. [Tesar, 2010 (3)], [Pandian, et.al, 2008][Mundst, et. al, 2005][Friedl and Allan, 2004]

Decision making regarding warfighter readiness represents a unique scenario as it involves inputs from multiple physical sensors as well as humans (the soldier and the commander). In this case, the subjective/qualitative judgment provided by the soldier or the commander may be construed as a sensor reading for an abstract parameter (which cannot be directly measured) such as the prevailing threat level in a combat situation. Thus the soldier and the commander may themselves be considered as 'abstract' higher-level sensors as they represent the only sources through which knowledge of abstract parameters can be obtained.

A wrong assessment of performance capability due to one or more faulty sensors (physical or abstract) may result in sub-par soldier performance. For instance, if the heart rate sensor indicates a very high reading, the soldier may be required to cease all activity until the value falls to an acceptable range but the existing combat situation may demand that the soldier execute a strenuous maneuver. The challenge facing the commander before making a call as to whether or not to instruct the soldier to proceed with the maneuver is to determine if the high reading is due to a faulty heart rate sensor or due to excessive stress faced by the soldier due to other factors. The following section now describes an example of how the SPFDI algorithm (Section 5.4) may be used to effectively utilize the information from all the available sensors to help resolve such operational conundrums[17].

[17] [Borsotto, et. al., 2004]present another application of Bayesian networks for health assessment of a soldier based on the data obtained from a suite of physiological sensors worn by the soldier. The objective of their work was primarily to determine the Presence, Absence or Unknown (PAU) state of life-signs and communicate it to the medics who can then prioritize the urgency of medical care needed or recommend an evacuation of the soldier from the battlefield.

Suppose the task performance (*TP*) level of soldiers for some task (say, shooting) is evaluated by a field commander as {good, acceptable, bad} based on their cognitive ability (*CA*), as perceived by the commander (say, on a scale of 1 to10, from low to high). The prevailing cognitive ability of a soldier is in turn, dictated by a host of internal and external factors. The threat level (*TL*) perceived by the soldier and the rest condition (*RC*) or sleep deprivation have a significant effect on the brain neural activity indicated by the Electroencephalography measure (*EEG*). A high perceived *TL* and low *RC* (signifcant sleep deprivation) may lead to a reduced level of *EEG* and hence *CA*. In addition, the combat load (total weight of the equipment) carried by the soldier (*SL*) and the ambient temperature (*AT*) play a significant role in regulating the soldier's core body temperature which in turn influences the soldier's *CA*. For instance, elevated values of *SL* and *AT*, may contribute to a higher value of *BT* and again lead to reduced level of *CA*. Figure 6-8 shows the representative causal Bayesian network that may be developed by a domain expert (for instance, based on the work of [McFarland and Tesar, 2010]). Table 6-14 shows an example of an instantiation table for this network.

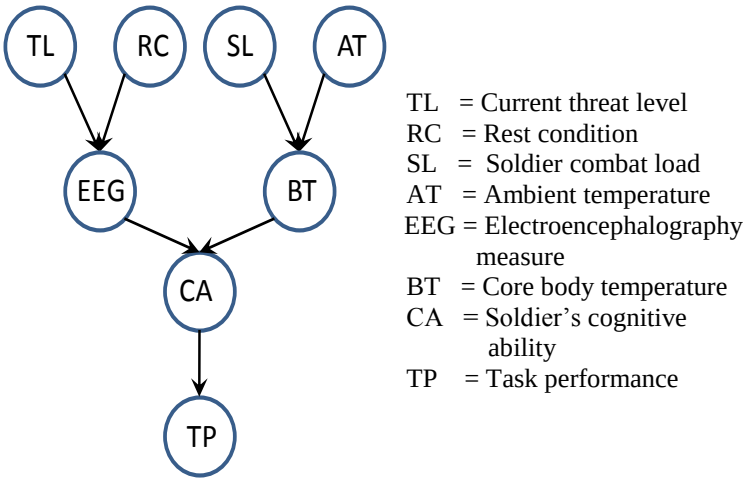


Figure 6-9: Representative Bayesian Network for Warfighter Task Performance Monitoring

Consider a situation where the soldier is engaged in a combat and is under heavy fire. The commander may be unable to visually observe a soldier and the only information available to the commander is the data from the soldier's sensor suite and communication with the soldier. Based on the communication, the commander may judge the soldier's *CA* and *TP* to be in the favorable range. If in this case, all the other sensors

and processes represented in Figure 6-9 are operating correctly, then starting from a value of 0.5, all the W_S and W_P belief values would attain a value of 1 when the SPDFI algorithm is executed based on the instantiation table.

Table 6-14: Example of an Instantiation Table (for Figure 6-9)

TL	RC	SL	AT	EEG	BT	CA	TP	TL-> EEG	RC-> EEG	SL-> BT	AT-> BT	EEG-> CA	BT-> CA	CA-> TP
NI	*NI*			*NoI*				*PoI*	*PoI*					
NI	*NI*					*NoI*		*PoI*	*PoI*			*PoI*		
NI	*NI*						*NoI*	*PoI*	*PoI*			*PoI*		*PoI*
		NI	*NI*		*NoI*					*PoI*	*PoI*			
		NI	*NI*			*NoI*				*PoI*	*PoI*		*PoI*	
		NI	*NI*				*NoI*			*PoI*	*PoI*		*PoI*	*PoI*
NI	*NI*	*NI*	*NI*			*NoI*		*PoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*	
NI	*NI*	*NI*	*NI*				*NoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*
				NI	*NI*	*NoI*						*PoI*	*PoI*	
				NI	*NI*		*NoI*					*PoI*	*PoI*	*PoI*
						NI	*NoI*							*PoI*

Table 6-15: W_S and W_P Values with No Sensor/Process Faults (for Figure 6-9)

TL	RC	SL	AT	EEG	BT	CA	TP
1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

TL->EEG	RC->EEG	SL->BT	AT->BT	EEG->CA	BT->CA	CA->TP
1.000	1.000	1.000	1.000	1.000	1.000	1.000

In this case, though the values for variables such as *TL*, *SL*, *EEG*, *BT* may be high, the commander can be assured that the soldier still maintains good cognitive ability to retain full situational awareness and provide maximum performance in the combat situation (increased chances of survivability). Now consider a combat scenario where the soldier confirms a high threat level *TL* and the *RC* sensor reading confirms that the soldier is well rested, but the commander notices that the readings received from the *EEG* sensor are unusually out of range, indicating neural activity not commensurate with the *TL* and *RC* values. This may potentially indicate a reduced cognitive awareness (*CA*), which may in turn make the soldier more vulnerable to enemy fire. However, based on communication with the soldier, the commander may judge the values of *CA* and *TP* to also be still in the acceptable range, thus resulting in contradicting assessments of soldier

capability. Suppose the W_S and W_P belief values in this case are as follows:

Table 6-16: W_S and W_P Values with a Sensor Fault (for Figure 6-9)

TL	RC	SL	AT	EEG	BT	CA	TP
0.800	0.800	1.000	1.000	**0.000**	0.3333	0.800	0.800

TL->EEG	RC->EEG	SL->BT	AT->BT	EEG->CA	BT->CA	CA->TP
0.800	0.800	1.000	1.000	0.666	0.666	0.800

A W_S belief value of 0.000 for the *EEG* sensor indicates that there may be a potential fault in it but higher belief values for all the other sensors and processes indicates that everything else is operating correctly. If the same pattern is observed after multiple iterations of the SPFDI algorithm, then the commander can be assured that the soldier still retains good cognitive awareness and the unusual *EEG* readings could be the result of a faulty electrode in the sensor and not because of abnormal neural activity.

Now under a similar combat condition, suppose the commander perceives that the *CA* for the soldier is low and the *TP* value is not up to the mark either. However, the readings from all the sensors indicate normal values and the *TL* as perceived by the soldier is also not significant, implying a high value of *CA* and hence normal/acceptable values of *TP*. Once again, the commander needs to be able resolve the contradiction whether the subjective values assessed by the commander are flawed or if there is more than one faulty sensor in the suite worn by the soldier (including a wrong *TL* assessment by the soldier). Suppose the W_S and W_P belief values from the SPFDI algorithm in this case are as follows:

Table 6-17: W_S and W_P Values with a Process Fault (for Figure 6-9)

TL	RC	SL	AT	EEG	BT	CA	TP
0.200	0.200	0.200	0.200	0.333	0.333	0.200	0.200

TL->EEG	RC->EEG	SL->BT	AT->BT	EEG->CA	BT->CA	CA->TP
0.200	0.200	0.200	0.200	**0.000**	0.333	0.400

Based on the W_S and W_P belief values in the table above, it is seen that a process fault is likely in the *EEG->CA* process (a definite fault may be confirmed after a few

iterations of the SPFDI algorithm). This indicates that the mapping between the neural activity as indicated by the *EEG* to the cognitive awareness of the soldier has changed significantly. The reduced cognitive awareness could be the effect of post-traumatic stress from an earlier mission that remained undetected in the initial assessment of the soldier capability or the commander was unaware of before the soldier was deployed on the mission. With this knowledge, the commander may decide to re-assign the soldier to a different role in order to protect both the individual soldier as well as others working alongside, since a reduced *CA* may make the soldier more liable to commit errors that may compromise the success of the mission. Since the soldier possesses a variety of skills, the commander may re-assign the soldier to a task where the prevailing value of *CA* for the soldier may be acceptable, in lieu of the soldier's other capabilities, to produce the desired level of *TP*.

6.7. CHAPTER SUMMARY

The goal of this chapter was to examine the applicability of the SPFDI algorithm developed in Chapter 5 in a variety of application domains, in order to demonstrate its usefulness in different scenarios. Four broad application classes, each consisting of a complex system equipped with multiple sensors for tracking different variables of interest in that domain, were considered for this purpose: a) applications where human safety is of paramount importance, b) applications where availability of the monitored system is the dominant requirement, c) applications with a high cost of system failure, and d) applications where performance maximization is the principal objective. The common thread linking all these application classes is the involvement of a human operator in the operational decision making loop. As observed in the different sections of this chapter, the function of the belief values obtained using the SPFDI algorithm is to aid the human operator in making the appropriate operational decision choices in each of the applications. Three representative examples were presented for each class of applications, with an in-depth description (development of a representative Bayesian network, creating an instantiation table, demonstration of a sensor fault and a process fault) of a selected example in each class, as summarized in Table 6-18.

Table 6-18: Chapter Summary

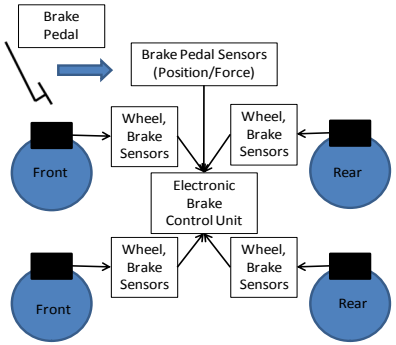
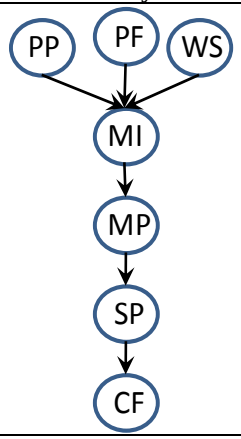
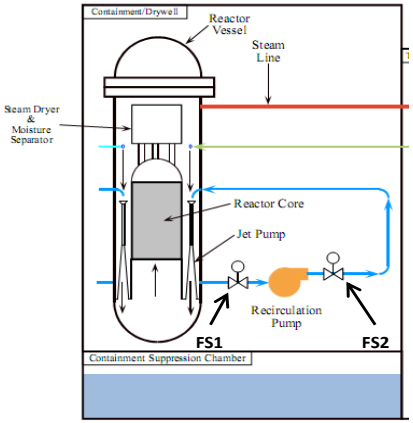
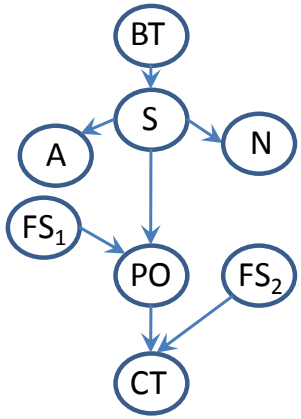
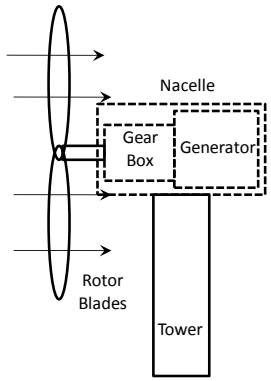
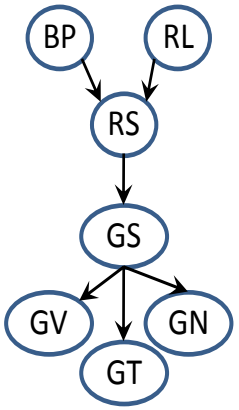
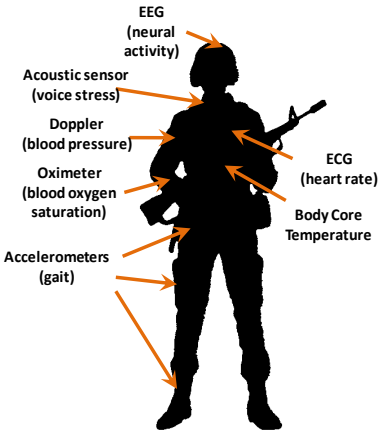
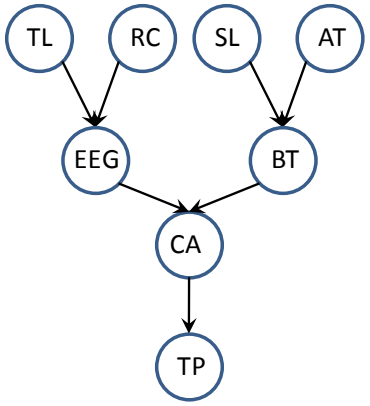
	Class	Example Applications	Representative Bayesian Network	Notation
1	Human safety at stake (Section 6.3)	Automobile brake-by-wire system 		*PP* =Pedal position sensor *PF* = Pedal force sensor *WS* = Wheel speed sensor *MI* = Motor current sensor *MP* = Motor rotor position sensor *SP* = Ball screw/brake caliper linear position sensor *CF* = Caliper force sensor
2	System availability (Section 6.4)	Nuclear power plant coolant circulation system 		*B T* = Bearing temperature *A* = Accelerometer (vibration) *N* = Acoustic noise *S* = Pump operating speed *PO* = Volumetric pump output *CT* = Reactor core temperature *FS1*, *FS2* = Flow sensors

Table 6-18: Chapter Summary

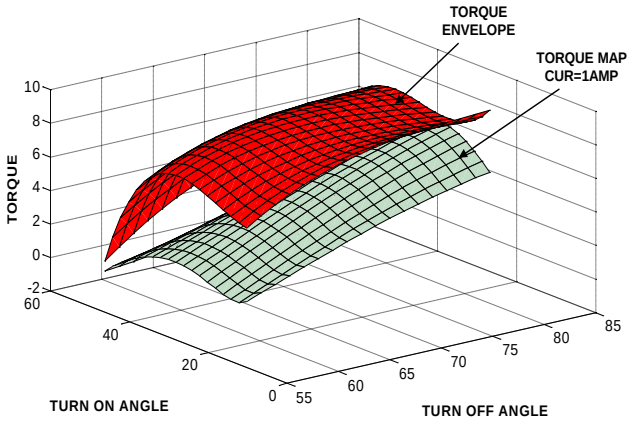
3	High cost of system failure (Section 6.5)	Wind turbine gearbox  	 	*BP* = Blade pitch angle *RL* = Wind load acting on the rotor *RS* = Rotor speed = Gearbox input speed *GS* = Gearbox output speed = Generator operating speed *GV* = Gearbox vibration amplitude *GT* = Gearbox temperature *GN* = Acoustic noise generated by the gearbox
4	Performance maximization (Section 6.6)	Warfighter task performance monitoring  	 	*TL* = Current threat level *RC* = Rest condition *SL* = Soldier combat load *AT* = Ambient temperature *EEG* = Electroencephalography measure *BT* = Core body temperature *CA* = Soldier's cognitive ability *TP* = Task performance

Chapter 7. Summary and Recommendations

This goal of this chapter is to provide an executive summary of the research work presented in Chapters 1 to 6. Sections 7.1-7.5 of this chapter review the research objectives, details of the research methodology, the original contributions and results from this work. Based on the understanding gained from the research findings, Section 7.6 of the chapter outlines some directions to be pursued for future research.

7.1. RESEARCH OBJECTIVES

The University of Texas Robotics Research Group (UT-RRG) has helped create the foundation for an open architecture of intelligent, reconfigurable systems with the intelligent Electro-Mechanical Actuator (EMA) at its core. An intelligent EMA has complete awareness of its operational capability at all times and the ability to reconfigure its available resources optimally to adapt to varying operational duty cycles or task requirements. The essential requirement of complete operational awareness is met through two key sources of information regarding the EMA capabilities. The primary source is comprehensive parametric models of the performance capability of a new (as-built) EMA over the complete range of its operating points, represented in the form of decision surfaces (performance maps and envelopes), used in conjunction with user-defined/application-specific performance criteria [Tesar, 2009] (Chapter 1, Section 1.2.1). The decision surfaces embody knowledge in a feedforward sense i.e. they represent how an EMA can be expected to perform under specific control parameter conditions over its entire operating



[Ashok and Tesar, 2007]

Figure 7-1: Decision Surfaces for Intelligent EMAs

range. To complete the decision cycle, this knowledge needs to be updated with reliable and accurate feedback information reflecting the actual existing conditions of the EMA (for control as well as for implementing Condition Based Maintenance (CBM) algorithms to track the health of the various EMA sub-components). This can be achieved by using a suite of multiple, in-situ sensors (Chapter 1, Section 1.2.2 and Appendix A). The combination of decision surfaces and real-time sensor feedback is used in a human-in-the-loop approach to operational decision-making where decisions are made based on residuals between the decision surfaces and sensor-inferred values (Chapter 1, Section 1.3). The supervisory role and operator judgment based on experiance enables more effective use of EMA resources in dynamic environments [Tesar, 2010 (4)].

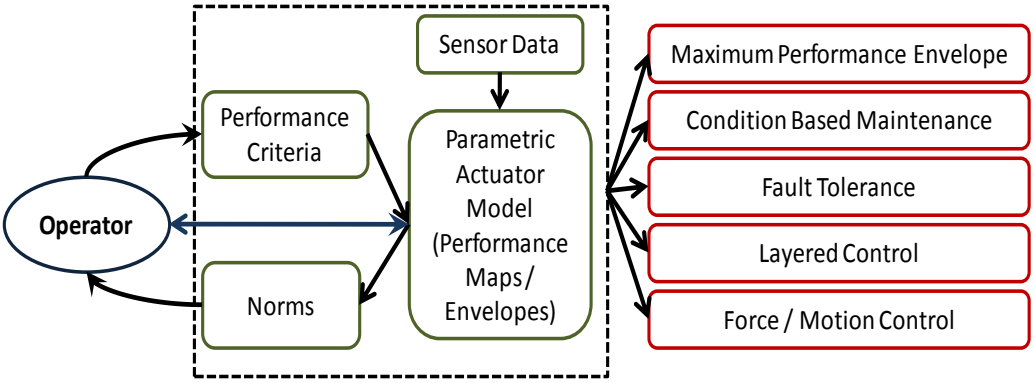


Figure 7-2 : Decision making in Intelligent Electromechanical Actuators

The quality of any human-in-the-loop decisions will only be as good as the information available to the decision maker. The available sensor data forms the basis for generating all the decision surfaces as well as in real-time evaluation of the EMA capabilities. The use of any erroneous data in updating the decision surfaces may result in corruption of these surfaces, leading to incorrect operational decisions being made by the operator or resulting in false alarms being raised during condition monitoring. It may also result in unpredictable system behavior with potentially damaging consequences. *Accurate and dependable sensor data is thus of vital importance*.

Many factors, both internal and external affect the quality, availability and the use of sensor data viz. noise, incorrect calibration, wear and degradation of the sensing

element during operation, temperature, shock/vibrations, uncertainty introduced as a result of the inherent sensor physics or during the data acquisition process, partial or complete sensor malfunction caused due to a failure of the sensor itself or due to connector/wiring harness failures, etc. (Chapter 1, Section 1.4). But operational decisions still have to be made despite imprecise and uncertain sensor data. The objective of this work is therefore to develop computational tools to ensure 'correctness' and usefulness of data and utilize information from all the sensors cohesively to provide information of greater value that can be used by the system operator for both control and diagnosis.

Failure to identify an abnormal sensor or sub-system and take appropriate corrective action could result in expensive and unnecessary system shutdowns or, worse, accidents that endanger both system and personnel. Hence, some safeguard is needed to provide advanced warning of any form of sensor or system degradation or impending failure. With any unwarranted deviations in sensor readings from their expected behavior, the challenge is to determine if the variation indicates a change in the states of the monitored EMA or if it is the result of one or more malfunctioning sensors.

The underlying assumption in many approaches used for sensor fault detection is that the monitored system is operating normally and the unusual readings are purely due to sensor faults. However, there may be cases where the fault does not lie in the sensor and its atypical signal content is caused by changes in the monitored system (or any of its components). On the other hand, fault detection or condition based maintenance algorithms used at the system level, while estimating the existing condition of the system, make the implicit assumption that all the sensors that are monitoring the system are operating correctly. In such cases, using data from sensors with faults can result in incorrect estimates of the monitored system's capabilities, its estimated health or remaining useful life.

The conundrum is thus to distinguish between these two distinct scenarios without making any assumptions regarding the condition of the sensors or the system and identify the precise source of abnormality in sensor readings. In order to address the above issues, the following research objectives were identified (Chapter 1, Section 1.5)

Research Objectives:

- Create a unified framework that can detect any system faults and sensor faults concurrently, independent of any assumptions regarding the health/condition of either the sensors or the monitored system (Chapter 5).
- Use an empirical data-driven approach to functionally correlate the variables of interest in an EMA in a physically meaningful manner (Chapters 3).
- Utilize well-defined mathematical principles to quantify, combine and propagate uncertainties resulting from different facets of the system operational decision making process, in a computationally tractable and scalable manner (Chapter 2).
- Provide a visual representation of the relationships between the different physical variables to complement performance maps as well as preliminary quantitative indicators for the condition of different system components and sensors in order to enable a human-in-the-loop approach to operational decision making for optimal allocation of the available resources (Chapters 4 and 6).

7.2. MODELING USING BAYESIAN NETWORKS

To accomplish the aforementioned research goals, an inclusive model is needed to succinctly capture the interaction among different EMA parameters, using all the available measurands in a cohesive fashion. The model should allow assimilation of existing domain knowledge, with easy interpretability to enhance human understanding of the model as well as the confidence to use it during system operation. It should support quick evaluations (forward/inverse) and allow for easy updates when new information becomes available. In particular, its ability to account for uncertainty in data is important. Although conventional engineering analysis tends to be deterministic, in reality, every variable has some uncertainty associated with it; inherently introduced as a result of simplifying assumptions made during a modeling process as well as during measurement.

Different methods exist to quantify uncertainties such as probability theory, fuzzy logic, Dempster-Shafer belief functions, etc. Each of these methods has been successfully used in different domains and each may be more suitable to a particular domain/objective than others (Chapter 2, Table 2-1). For the present work, since decision making for

intelligent EMAs is based on extensive empirical data (performance maps), a probabilistic approach has been deemed to be most suitable. This enables the system model to be updated regularly in dynamic environments and also encapsulates a large set of operating conditions in a concise manner. Both these factors are important for intelligent EMAs where performance maps need to be periodically updated to reflect the extant conditions.

The use of Bayesian networks to model the intricate relationships among the variables in intelligent EMAs is proposed here. This approach is data-driven, provides an intuitive presentation based on physical principles and utilizes probability theory (Chapter 2, Section 2.3) for quantifying, combining and propagating uncertainties. It also provides functional redundancy for all the variables involved. The graphical nature of the technique provides a greater insight into the observed system behavior whereas in many other approaches such knowledge is not very easily accessible. The resulting model thus has the advantage of being easily used when humans are involved in the decision making process, as is the case in intelligent EMAs. The approach also complements the overall EMA decision-making methodology developed in [Ashok and Tesar, 2007].

A Bayesian network is a Directed Acyclic Graph (DAG) structure that consists of a set of nodes that represent discrete/continuous random variables $X = \{X_1, X_2, \ldots X_n\}$ pertinent to the domain. These nodes are connected in pairs by directed links that represent the probabilistic dependencies between the variables. The lack of an arc between two nodes explicitly represents their independence. The strength of the relationships and the inherent uncertainties in the relationships between the nodes is quantified using conditional probability distributions, encoded as Conditional Probability Tables (CPT) at each node. [Pearl, 1988] defines such a set of variables and a joint probability distribution defined over all the variables as a '*probabilistic model*'.

Developing a Bayesian network for a given application involves three main steps:

- Identifying the pertinent variables to be represented as nodes along with their possible states that they may assume.
- Identifying relationships between the variables and representing them graphically

by connecting the appropriate nodes with the links directed in a specific manner

- Quantitatively characterize these relationships i.e. populate the CPTs for all the nodes in the network with the appropriate probability distributions.

In the present work, a knowledge representation approach based on extensively available domain knowledge has been used to develop the qualitative structure of the Bayesian network (see Chapter 3, Sections 3.3.1-3.3.8). The parameterization of the different node CPTs is done using the same empirical data set used to generate of performance maps (Chapter 2, Section 2.4.3.2).

Once the structure of the network has been finalized, the process of determining the updated posterior probability distributions of the nodes in the network with the knowledge of a subset of those nodes (by conditioning on observed values), is referred to as belief updating or probabilistic inferencing [Korb and Nicholson, 2004]. Essentially, this process allows for propagation and combination of uncertainties associated with the different variables. The nodes whose values are known or observed are called the *evidence* nodes. The subset formed by one or more nodes (other than the evidence nodes) whose posterior distribution values are calculated are referred to as *query* nodes.

With simple causal chains the process of belief updating can be performed using a combination of the chain rule of probability and Bayes' theorem. However, for more complicated networks, various algorithms have been developed by researchers for belief updating. These algorithms can be broadly classified into two categories as exact and approximate inferencing algorithms. The choice of an appropriate algorithm is determined by factors like the structure of the Bayesian network (singly or multiply connected), available computational resources, timing constraints, etc. Three popular algorithms available in the literature were explored in detail in this work. All three algorithms have also been implemented as part of the AMOS C++ library (Appendix C).

The first algorithm studied was the Pearl's belief propagation algorithm [Pearl, 1988], an exact inferencing algorithm to propagate beliefs in polytree networks (Chapter 2, Section 2.5.2.1.1). The algorithm makes use of the local independencies between the nodes as embodied by the structure of the network and Bayes' theorem to determine the

posterior probability distributions of the query nodes. To do this, two types of parameters, λ and π values, are used in the conjunction with the CPT of a node to update its belief. The λ and π values are calculated based on two types of messages sent and received by every node in the network. The λ messages are sent from a child node to all its parent nodes. The π messages are sent from a parent node to all its child nodes.

In addition, two stochastic simulation algorithms for approximate inferencing were also studied, viz. the Logic Sampling algorithm (Chapter 2, Section 2.5.2.2.1) and the Likelihood Weighting algorithm (Chapter 2, Section 2.5.2.2.2). In these algorithms, inferencing is accomplished by sampling data for a number of trials based on the conditional probability distributions of the different nodes. The generated samples are used to determine the posterior probability distributions of the nodes of interest. As more and more cases are generated, the estimated posterior distributions of the different nodes converge to increasingly exact distributions based on the law of large numbers (based on user-defined convergence criteria for an application). The main drawback of the Logic Sampling algorithm is that it rejects cases for which the samples generated for evidence nodes are inconsistent with the observed values (especially when the probability of those values occurring is low). This leads to slower convergence times. To mitigate this issue when dealing with unlikely evidence values, and to make use of every generated sample, the Likelihood Weighting algorithm utilizes the fractional likelihood of occurrence of the evidence values (called as '*score*') while estimating the posterior probabilities.

Table 7-1: Modeling of Intelligent EMA using Bayesian Networks

Topic	**Chapter/ Section**	**Description**
Modeling of Intelligent EMAs	Section 2.2	Requirements of modeling technique • Ability to accurately and compactly represent systems with a minimum of simplifying assumptions • Incorporate existing domain knowledge (available data) • Intuitive to understand • Support forward and inverse decision-making • Allow incorporation/ propagation of uncertainty in system parameters • Fast evaluation and modification/reconstruction

		• Adaptability to rapidly changing environments
Bayesian Networks	Sections 2.3, 2.4 and 2.5	• Graphical modeling using Directed Acyclic Graphs (DAGs) • Based on well-developed concepts of probability theory • Compact representation of the joint probability distribution of a group of random variables (discrete/continuous) • Explicit representation of probabilistic dependencies (nodes connected by directed links, characterized by conditional probability tables or CPTs) • Updated knowledge of values of different variables in the network based on a subset of observed/evidence values obtained via belief updating • Use of inferencing algorithms to update beliefs — Exact Inferencing (Pearl's Belief Propagation Algorithm) — Approximate Inferencing (Logic Sampling, Likelihood Weighting)

7.2.1. Bayesian Network for a Switched Reluctance Motor

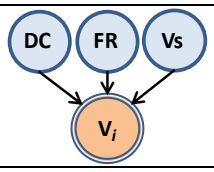
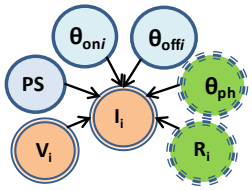
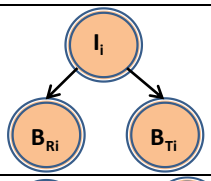
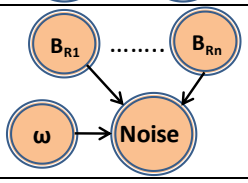
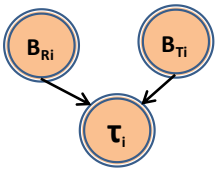
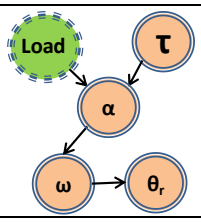
For many engineering systems, researchers explore specific aspects of the system in great detail while not accounting for other facets that may still play an important role in influencing the overall system behavior. However, when taken as a combination, these different fragments of knowledge can be used to develop a truly comprehensive model of the system that may be used for effective operational decision making.

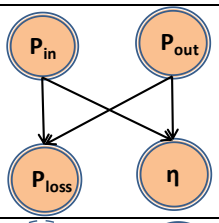
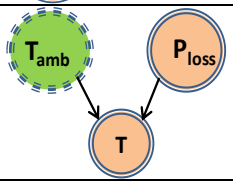
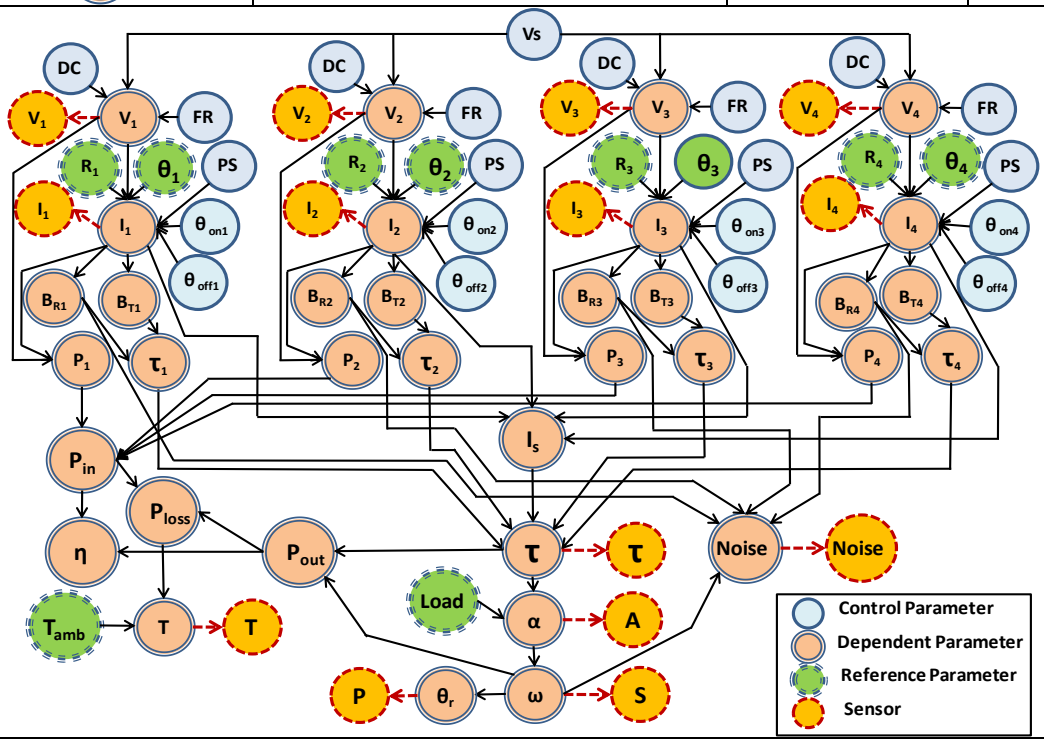
The primary component of the testbed in the laboratory at UT-RRG used in this work (Appendix B) is a four phase switched reluctance motor, the design and operational aspects of which have been studied widely by many researchers. Highly comprehensive and nonlinear models dealing with individual aspects of motor operation such as torque generation, generation of acoustic noise and vibrations, thermal characteristics, etc. are readily available in the literature and are widely used for both design and simulation purposes but there is a distinct paucity of comprehensive models that account for all these aspects simultaneously. The development of a detailed Bayesian network for the testbed was therefore pursued in Chapter 3 as a unified tool for graphical modeling of the interaction between the different physical phenomena occurring in the test system.

To develop the network in its entirety, various sub-models depicting the causal influences among the control parameters (supply voltage, PWM duty cycle and frequency for each phase, phase status and turn-on/turn-off angles), the reference parameters (phase resistances, relative phase angles, brake load and ambient temperature), and the dependent parameters (phase currents, radial and tangential flux densities, torque, speed,

acceleration, noise, losses, etc.) are linked together. The combined model, the different sub-models, and the relevant sections are shown in Table 3-1. The causal network provides a compact and physically relevant visual representation of the correlations between the variables in the system to the end-user. The uncertainty associated with each variable is automatically taken into account by the empirical data used to populate the node CPTs while inferencing algorithms help propagate and combine these uncertainties.

Table 7-2: Bayesian Network for a Switched Reluctance Motor

Sub-network	Notation	Description	Chapter/ Section
	i = 1…n = Number of motor phase, DC = PWM Duty Cycle, FR = PWM Frequency, V_s = Supply Voltage	Influence of Control Parameters on Motor Phase Voltage	Chapter 3, Section 3.3.1
	i = 1…n = Number of motor phase, V_i = Phase Voltage, R_i = Phase Resistance, Θ_{ph} = Rotor Angle Relative to Stator, PS = Phase Status (On/Off) Θ_{oni} , Θ_{offi} = Phase Turn-on/off Angles I_i = Phase Current	Relation between Phase Voltage, Phase Current and Phase resistance	Chapter 3, Section 3.3.2
	i = 1….n= Number of motor phases, I_i = Phase current B_R = Magnetic flux density, radial B_T = Magnetic flux density, tangential	Relation between Current and Flux Densities	Chapter 3, Section 3.3.3
	i = 1….n = Number of motor phases, B_R = Magnetic flux density, radial ω = Motor Speed, N= Acoustic Noise	Relation between Radial Flux Density, Motor Speed and Acoustic Noise	Chapter 3, Section 3.3.4
	i = 1….n = Number of motor phases, B_R = Magnetic flux density, radial B_T = Magnetic flux density, tangential τ_i = Torque produced by a single phase	Relation between Flux Density and Torque	Chapter 3, Section 3.3.5
	τ = Total Electromagnetic Motor Torque α = Angular Acceleration of Rotor ω = Angular Velocity of Rotor θ_r = Rotor Position $Load$ = External Load Torque	Relation between Motor Torque, Acceleration, Speed and Position	Chapter3, Section 3.3.6

	P_{in} = Input Electrical Power P_{out} = Output Mechanical Power P_{loss} = Total Power Losses η = Motor Efficiency	Relation between Input Power, Output Power, Losses and Efficiency	Chapter 3, Section 3.3.7
	T_{amb} = Ambient Temperature P_{loss} = Total Power Loss T = Average Motor Temperature	Relation between Power Losses and Motor Temperature	Chapter 3, Section 3.3.8
			

Detailed Bayesian Network Model of a Four Phase Switched Reluctance Motor Obtained by Combining the Smaller Sub-models (Chapter3,Section 3.2)

7.3. CRITERIA-BASED RESOURCE MANAGEMENT

In many applications it may not be feasible to either design/develop an exhaustive network as discussed in the previous section or utilize it during system operation. Numerous constraints can render such detailed networks impractical for use in operational decision making. These include factors that can influence:

- The physical design of the system itself —The factors in this category include the number and type of sensors needed to adequately respond to changes in the

system and its environment, the number of sensors that can be integrated without requiring significant design modifications to the system, the sensors that are absolutely critical for system operation, the type and degree of physical and/or functional redundancy that can be achieved using a particular set of sensors, etc.

- The features of the Bayesian network representation of the system —This category encompasses factors such as the number of nodes corresponding to the random variables of interest, the granularity of discretization (number of states) of each node, the type and the degree of interconnection between nodes, the size of CPTs for all the nodes, nodes representing variables that are measurable using the available sensors, nodes whose values can only be inferred, etc.
- The manner in which the Bayesian network is used for decision making during system operation— Included in this category are factors such as the nodes which may be set as evidence during inferencing, the condition and accuracy of sensors that are used to determine which node states can be set as evidence, the influence of different evidence nodes on different variables, the type of information that can be gleaned using the available network structure and evidence, etc.

Ideally, the design of a Bayesian network representation should be carried out in tandem with the physical system design. The process tends to be iterative as numerous constraints need to be balanced concurrently. Once the design of the physical system (with the requisite sensors) and its representative network is complete, the next step is to effectively manage the information that can be obtained while the system is in operation (for instance prioritizing a subset of sensors to address a specific objective, decide on alternative sensors to infer the value of a deteriorating sensor/s, address constraints like limited bandwidth/power, decide on inferencing algorithms best suited to meet the computational constraints, etc.)

There is an evident need for a criteria-based resource management methodology that can help decide upon and best utilize the available sensing resources at any given time. The goal is to define and use suitable criteria that can help provide guidelines to both the system designer and the end-user to use a finite set of sensors, empirical

data/decision surfaces, etc. in concert with available computational resources and maximize the utility of the information that can be obtained regarding the system at any given time. This can provide a synergy between different sources of information as well as influence the availability of information in situations where data from specific sensors becomes partially or completely unavailable. It can also help avoid overwhelming data transfer, computational and memory requirements in a multi-sensor system by ensuring that only the truly essential data is acquired and utilized. Different design criteria (Chapter 4, Sections 4.3.1-4.3.10) and operational criteria (Chapter 4, Sections 4.4.1-4.4.8) were proposed to address the objectives outlined above and are summarized in Table 7-3 and Table 7-4 respectively. These representative criteria are intended to motivate the definition of more application-specific criteria in the future.

Table 7-3: Design Criteria (for a system and its Bayesian network representation)

No.	Criterion	Description /Importance	Chapter/ Section
1	Application Requirements	Any application environment consists of sensors that are essential for operation as well as additional optional sensors. The total number of nodes in the Bayesian network is dictated by the number of sensors used in any application.	Chapter 4, Section 4.3.1
2	System Design Constraints	The number of sensors used in any application is dictated by their form factors, cabling, etc. that satisfy the size and volume constraints imposed by the application. These design constraints thus play a vital role in determining the size of the Bayesian network by influencing the sensors that can be actually used in the application.	Chapter 4, Section 4.3.2
3	Overall Costs	The overall costs associated with sensors and peripheral equipment like data acquisition hardware, power supplies, etc. may influence which sensors are integrated into a system and hence influence the number of nodes in the Bayesian network representation	Chapter 4, Section 4.3.3
4	Relative Importance of Sensors	The relative importance of a sensor or a subset of sensors may be used to determine the direction and number of the links between the nodes. This allows the manipulation of network topology to provide the desired degree of redundancy for the most critical measurands.	Chapter 4, Section 4.3.4
5	Type of Data Available	The availability of data in specific formats (for instance, due to experimental constraints) influences the manner in which the conditional independencies between variables can be represented (i.e. which nodes in the network may be linked as well as the direction of the arcs) and how the CPTs for different nodes can be populated	Chapter 4, Section 4.3.5
6	Causality	Causal relations between physical variables can be used to determine the node ordering, the direction of links and the level of interconnectedness in a Bayesian network. The use of causality can result in more practical, intuitive, physically relevant and compact networks.	Chapter 4, Section 4.3.6

7	Discretization of Data	The number of distinct states into which each node in the network can be discretized is driven by application requirements. The granularity of discretization in turn influences the quality of results obtained using the network, speed of execution of inferencing algorithms, memory required to store the CPTs, etc.	Chapter 4, Section 4.3.7
8	Sensor Reliability	All the sensors used in monitoring are not equally reliable (different susceptibility to temperature, shock/vibrations, etc.). The operational history for sensors may be used to direct links from nodes representing the most reliable sensors to ensure constant availability of information in case of a loss of sensors corresponding to critical measurands.	Chapter 4, Section 4.3.8
9	Memory Requirements	The size of CPTs grows exponentially in the number of node parents, degree of interlinking and the granularity of discretization of node states. The associated memory requirements must be considered to satisfy constraints on availability of resources (storage), real-time operation requirements, etc.	Chapter 4, Section 4.3.9
10	Intended Use	Modifications such as adding additional nodes or links (or pruning existing nodes/links) in a network structure may be needed to satisfy specific requirements for the application (for example, fault detection and isolation), provide greater intuitiveness to the end-user, etc.	Chapter 4, Section 4.3.10

Table 7-4: Operational Criteria (for a system and its Bayesian network representation)

No.	Criterion	Description / Importance	Chapter/ Section
1	Sensor Characteristics	Depending on the accuracy of the sensor used for a specific measurand, the operator may choose to use the value of sensor reading directly or infer the value of the measurand via the Bayesian using the readings from the other sensors as evidence.	Chapter 4, Section 4.4.1
2	Node Distance	The uncertainty in the inferred value of a variable is influenced by the number of intermediate nodes/links between the evidence node/s and the query node. In general, the greater the separation, the greater is the uncertainty in the inferred value.	Chapter 4, Section 4.4.2
3	Sensor Health Status	With any unforeseen variations (drift, bias, excessive noise) in the readings from a sensor or a subset of sensors, the system operator needs to be able to determine if the readings are indicative of one or more faulty sensors or if it is the result of normal changes in the system response.	Chapter 4, Section 4.4.3
4	System Health Status	Unanticipated deviations in sensor readings may not necessarily be due to a faulty sensor but may be indicative of a more serious, potential fault in one or more of the monitored system's sub-components. A preliminary assessment of system health is needed to enable the system operator to make this distinction.	Chapter 4, Section 4.4.4
5	Resource Availability	Different sensors may need to be used in different system operating regimes (to provide the value for a specific variable via direct measurement or by inferencing using the Bayesian network) in accordance with the bandwidth, sampling rates, etc. that are possible with the available system hardware.	Chapter 4, Section 4.4.5
6	Strength of Relationship between Nodes	Some variables/nodes in the network may be more strongly correlated (or may have a direct causal influence) as compared to others. The strength of these relationships (quantified using the concepts of link strength and connection strength) may be used to	Chapter 4, Section 4.4.6

		determine which set of nodes are more appropriate for use as evidence for a particular set of query variables.	
7	Type of Query	The topology of the Bayesian network along with the conditional probability distributions of the different nodes represents a comprehensive database for a system which can be queried in multiple ways (probability of evidence, prior and posterior marginals, *MAP*, *MPE*) to obtain different types of information regarding the system and the sensors monitoring it.	Chapter 4, Section 4.4.7
8	Computational Complexity	Based on the network topology, discretization of the nodes, available computational resources, real-time operation requirements, etc. the operator may decide on the appropriate exact or approximate inferencing algorithm to use, number of samples to be used for stochastic sampling (approximate) algorithms, etc.	Chapter 4, Section 4.4.8

7.4. SENSOR AND PROCESS FAULT DETECTION AND ISOLATION

Evaluating the trustworthiness of available sensor data can provide guidance in many operational decisions. If it can be determined that a deviation in a sensor's reading is due to some issue with the sensor, then the operator can choose to ignore the data from that sensor for control purposes or to update a map. On the other hand, if it can be determined that the deviation is due to an issue with the system or its components, it could serve as additional knowledge of use for the higher level CBM algorithms (e.g. localizing the fault to a specific component) and the sensor readings can still be used to update the stored maps to accurately reflect the extant system status. The situations above are termed as '*sensor fault*' and '*process fault*' respectively and defined as follows.

The term '*sensor fault*' in the context of this work indicates disagreement between the ideal value that a sensor is supposed to indicate under the prevalent operating conditions and the measured value it actually indicates at the sampling instant under consideration. The output has to be tracked over multiple sampling instants to declare with certainty that the sensor is faulty. The link between every pair of nodes in a Bayesian network can be considered as a *process* that converts the physical parameter represented by the parent node to the one represented by the child node. The correlation between the two is quantified by the CPT of the child node. If there is simply a change in the operating conditions of the system, the conditional distribution for the child node given the value of its parents would still hold valid. But a fault in any system component would alter the physical relations among the variables associated with that component. In

other words, this would essentially render the relation encoded in the CPT of the child node invalid. This is termed as a '*process fault*'.

7.4.1. The Sensor and Process Fault Detection and Isolation (SPFDI) Algorithm

The principal focus of this work was to use the Bayesian network framework developed in earlier chapters to help distinguish between sensor and process faults concurrently. Although the use of Bayesian networks is more widespread in the area of fault diagnosis, different researchers have explored its use for fault detection and isolation as well. *Due to the limitations posed by various existing approaches (Table 7-5), a novel algorithm termed as Sensor and Process Fault Detection Isolation (SPFDI) algorithm has been developed in this work and represents the most significant contribution of this research*.

The value for any variable deduced using the Bayesian network represents the value that should ideally be obtained from its corresponding sensor if there are no unknown problems in either the sensors or the system components that have not already been accounted for when the system model (decision surfaces) was last updated. On the other hand, the values indicated by sensors at any sampling instant indicate the extant system status at that instant and will be influenced by any possible issues that have occurred since the maps were last updated. The premise of the SPFDI algorithm is that by sequentially instantiating different nodes in the network ('*Nodes Instantiated*' or *NI*), performing inferencing, and examining the resultant values of specific nodes in the network ('*Node of Interest*' or *NoI*), it is possible to examine the validity of the sensor readings obtained and distinguish between sensor and process faults.

Table 7-5: Literature Review of Existing Approaches using Bayesian Networks for Sensor Fault Detection and Isolation

Method	Features	Chapter/ Section
[Bickmore, 1994]	**Potential advantages**: • Realtime operation and ability to handle large number of sensors • Provides analytical redundancy, well-defined error rates **Potential disadvantages**: • Requires invariant analytical relations between all variables of interest	Chapter 5, Section 5.3.3.1

	• Cannot robustly handle multiple sensor failures • Distinction between sensor/system faults only with multi-point sensor failure	
[Aradhye, 1997]	**Potential advantages**: • Ease of implementation (continuous/discrete data, steady state/transient) • Unified framework for fault detection (sensor as well as system) as well as classification (normal, bias, precision degradation, complete failure) • Allows incorporation of system specific qualitative knowledge **Potential disadvantages**: • Explicit modeling of faults (with known effects on variables of interest) required to detect as well as distinguish between sensor and system faults • One-to-one mapping of operating modes for fault classification not realistic • Potential for false alarms due to rigid definition of fault modes	Chapter 5, Section 5.3.3.2
[Mehranbo-d, et. al., 2003]	**Potential advantages:** • Ease of implementation (steady state/transient conditions) • Threshold setting, fault detection/identification index, fault classification • Ability to handle multiple sensor faults **Potential disadvantages:** • Assumes no variation in system condition (cannot detect system faults) • May result in large CPTs for variables that have a wide operating range • Thresholds determined for variables with spread out distributions (i.e. low probability values for different states) may be unreliable.	Chapter 5, Section 5.3.3.3
[Ibarguengo-ytia, et. al., 2003]	**Potential advantages:** • Anytime algorithm (pre-compiled binary decision tree) • Ability to detect multiple sensor faults under specific conditions • Ability to isolate faults (via quality measures based on information theory) **Potential disadvantages:** • Assumes no variation in system condition (cannot detect system faults) • Requires two separate networks • Model learned from data for specific operating conditions and may not be applicable over the entire operating regime of the system	Chapter 5, Section 5.3.3.4
[Mengshoel et. al., 2008]	**Potential advantages:** • Real-time operation (compiling Bayesian network into arithmetic circuits) • Proven for very large networks (hundreds of nodes) • Enables distinction between sensor and system faults **Potential disadvantages:** • Explicit modeling of faults required • Intricate network structures needed even for moderately complex systems	Chapter 5, Section 5.3.3.5

The readings from the sensors corresponding to *NI* are used to set specific states of such nodes as evidence to the network. Every node in the network is considered as a *NoI* sequentially one at a time, until all the nodes have been traversed. For each *NoI*, multiple values are estimated by considering different combinations of other nodes as *NI* and calculating its marginal posterior probability distribution. The values inferred for the *NoI* are then compared with the values indicated by their corresponding sensors to check if the sensors are indicating what they are supposed to under the prevalent conditions.

For each of the many *NI* and *NoI*, if the measured and inferred values concur (for the *NoI*), then it indicates that the system has not changed significantly from its last known condition. Hence, the assurance in the fact that the sensors are operating normally and relations among the variables ('*processes of interest*' or *PoI*) remain the same as before increases. Conversely if these values do not match, the assurance decreases.

By assigning a numerical measure to this level of assurance in different sensors and processes in the network, and incrementing or decrementing it suitably each time the measured and estimated values are compared for different *NoI* (increasing it when the values agree and vice-versa), it is possible to identify potential sensor faults. It can also help verify if the presumed conditional relations among the different variables, encoded in the CPTs of all the nodes still hold true and help identify process/system faults.

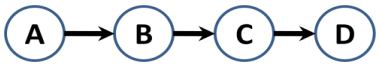


Figure 7-3: Bayesian Network

Table 7-6: Example of an Instantiation Table

	Sensors				Processes		
Step	A	B	C	D	A->B	B->C	C->D
1							
2							
3							
4							
5							
6							

Nodes Instantiated Node of Interest Process of Interest

The first step in implementing the SPFDI algorithm is to decide the instantiations of different nodes as *NI* and identify the *NoI* and *PoI* associated with each instantiation. In this work, the *NI* are always chosen from the set of ancestral/predecessor nodes for any *NoI*. An inclusive list of all such instantiations is referred to as the '*instantiation table*' for the network. The process of probabilistic propagation, considering each row of the table until all the rows have been exhausted is called a '*fault detection and isolation cycle*'.

Unique weights termed as 'belief values' are assigned to each node/sensor (W_S) and process (W_P) in the network, to indicate the confidence the operator has regarding their status as being faulty or not. The values are defined to lie in the interval [0, 1] with values tending to one implying perfect health and values tending to zero indicating a

potential fault. At the start of each new cycle, it is assumed that there is no knowledge of the status of sensors or processes. To account for this ignorance regarding the initial conditions, the beliefs for all the nodes and processes is initialized to a value of 0.5.

The amount ε_S and ε_P by which W_S and W_P are modified respectively, after each step in the fault detection and isolation cycle are determined by the number of times (n_S or n_P) a particular sensor or process figures either as a *NI, NoI* or *PoI* in the instantiation table and is considered a fraction of the initial weight for that sensor/process.

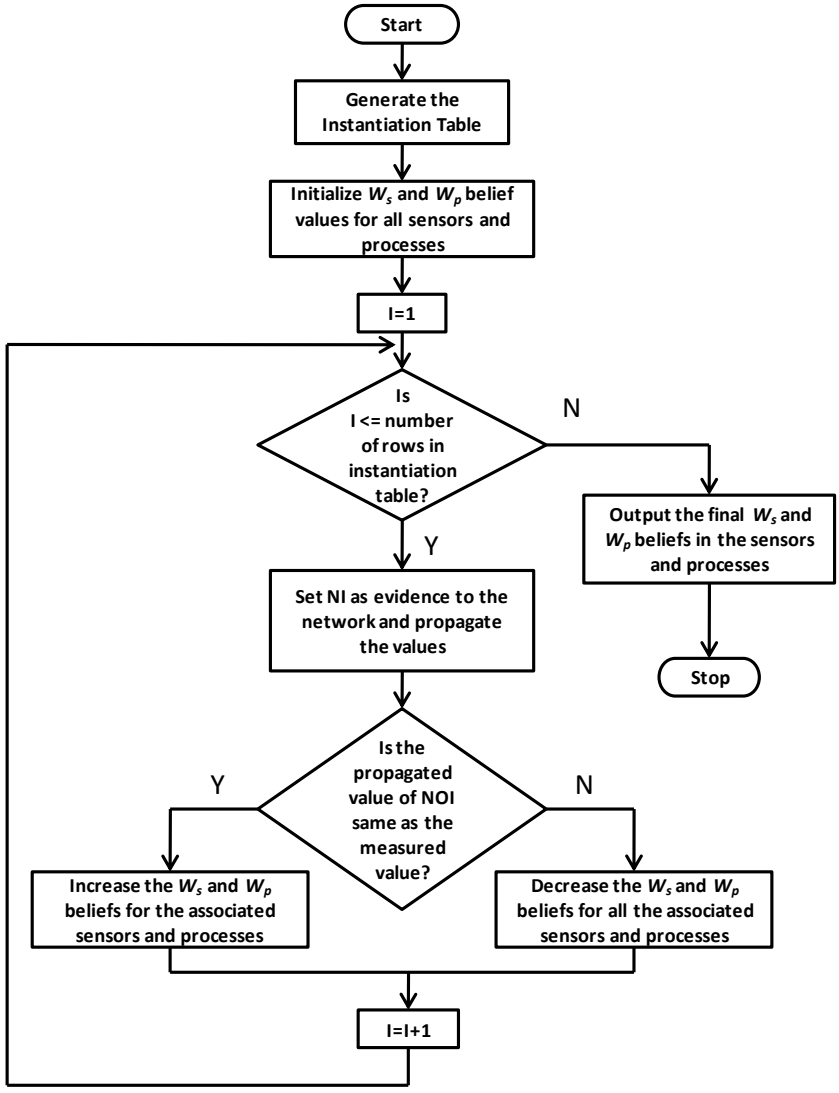


Figure 7-4: The Sensor and Process Fault Detection and Isolation (SPFDI) Algorithm

At the end of a single iteration, the sensor with W_{Smin} or the process with W_{Pmin} can be identified as being potentially faulty. Depending on the application, a suitable threshold may also be defined to indicate the lowest acceptable values for W_S and W_P, below which sensors or processes may be deemed faulty. This belief is further strengthened if the same results are obtained after multiple iterations of the algorithm.

The SPFDI algorithm is successful in isolating single sensor faults, process faults as well as multiple sensor faults, as shown in Table 7-7 to Table 7-9 for a simple hypothetical network as shown in Figure 7-3 (only a portion of the results are reproduced here, see Tables 5-3 to 5-5 for the complete results). However, certain special cases were identified where the algorithm produces incorrect or inconsistent results.

Table 7-7: Change in W_S and W_P Values with a Faulty Sensor

	Sensors				**Processes**		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	0.6667	**0.0000**	0.6667	0.6667	0.6667	0.5	0.6667

Table 7-8: Change in W_S and W_P Values with a Faulty Process

	Sensors				**Processes**		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	0.3333	0.3333	0.3333	0.3333	0.3333	**0.0000**	0.3333

Table 7-9: Change in W_S and W_P Values with Multiple Sensor Faults

	Sensors				**Processes**		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	0.3333	**0.0000**	**0.0000**	0.3333	0.3333	0.25	0.3333

With any unforeseen deviation in the values from a sensor corresponding to root or leaf nodes in the network, applying the algorithm results in the associated sensor as well as all the processes attached to that node. An explanation for this inconsistency was provided by identifying the '*subset problem*' (Section 5.5.3.1). In Table 7-6, where the

shaded cells in $Col_{A->B}$ and Col_A are the same, it can be said that $Col_{A->B} \subseteq Col_A$. Such overlap results in W_A and $W_{A->B}$ to be incremented or decremented by the same amount throughout the cycle. Thus, a subset may be formed between the columns corresponding to a root node and a process originating from that node, when the root node is considered as *NI* and its descendents are sequentially considered as *NoI,* with the process always included in the path between the *NI* and the *NoI* at every instantiation.

Similarly, it is observed that $Col_{C->D} \subseteq Col_D$ i.e. a subset may be formed between the columns corresponding to a leaf node and a process terminating at that node, when the leaf node is considered as *NoI* and its different ancestral nodes are considered as *NI,* with the process again always included in the path between the *NI* and the *NoI* at every instantiation. There may be other cases when columns corresponding to certain sensors or processes in the network form a subset of the columns that correspond to other sensors or processes that occur in the network, either individually or by considering a union of those columns. For instance, in Table 7-6, it is seen that $Col_D \subseteq (Col_C \cup Col_{B->C})$. In the above cases, if a real fault occurs in sensors/processes whose columns form the superset of columns corresponding to other sensors/processes in the network, then the SPFDI algorithm also incorrectly flags the sensors or processes corresponding to the columns in the subset as faulty. These situations are collectively termed as the '*subset problem'*.

Table 7-10: Change in W_S and W_P Values with a Fault in a Sensor Corresponding to a Root Node

	Sensors				**Processes**		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	**0.0000**	0.6667	0.6667	0.6667	**0.0000**	0.625	0.6667

Table 7-11: Change in W_S and W_P Values with a Fault in a Sensor Corresponding to a Leaf Node

	Sensors				**Processes**		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	0.6667	0.6667	0.6667	**0.0000**	0.6667	0.6667	**0.0000**

Table 7-12: Change in W_S and W_P Values when Columns in the Instantiation Table Form Subsets (resulting in incorrect conclusions of faults which do not exist)

	Sensors				Processes		
Step	**A**	**B**	**C**	**D**	**A->B**	**B->C**	**C->D**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
6	0.3333	0.3333	**0.0000**	**0.0000**	0.3333	**0.0000**	**0.0000**

The variations in the W_S and W_P values in these situations is shown in the tables above (only a portion of the results are reproduced here, see Tables 5-6, 5-7 and 5-9 for the complete results). To mitigate the complications created by the subset problem, it is desirable to eliminate the possibilities of subsets being formed as far as possible (especially if a particular sensor or a process is critical from the application point of view). A simple approach to do this was proposed by using additional nodes representing redundant sources of information (representing additional sensors, expert opinion, etc.) for the root and leaf nodes to distinguish between sensor and process faults (Section 5.5.4). The effect of adding these additional nodes to the network adds additional possibilities for the *NI* and *NoI* thereby resulting in additional rows in the instantiation table which could potentially make every column of the instantiation table distinct.

The two processes A_2->A and D_2->D do not represent real physical processes and are considered to be 'perfect' processes which never fail (the W_P belief values for such processes are always set to 1 during the execution of the SPFDI algorithm). With additional sensors and intermediate processes, the list of additional instantiations that can now be added are shown in Table 7-13. Note that this makes the individual columns distinct. From Table 5-11, it can be seen that the SPFDI algorithm can now successfully isolate the fault in the sensor corresponding to the node A without misdiagnosing a fault in the A->B process (as demonstrated in Table 5-6).

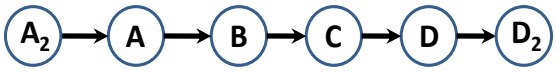


Figure 7-5: Addition of Redundant Nodes

The application of the SPFDI algorithm thus developed to the test set-up was then studied in Section 5.6. Figure 7-6 represents a Bayesian network of the test set up used in

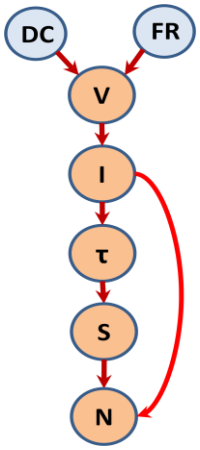
this work (Appendix B). It represents a simplified version of the detailed network developed in Chapter 3, using some of the criteria presented in Chapter 4 (such as the sensors available, memory and computational requirements, etc.). Figure 7-7 shows the basic network augmented with additional nodes to overcome the subset problem. Table 7-15 shows the instantiation table for the original network and Table 7-16 to Table 7-19 show the variation in the W_S and the W_P values with different types of sensor and process faults (only a portion of the results are reproduced here, see Tables 5-13, 5-15, 5-16 and 5-18 for the complete results). Table 7-20 shows the instantiation table for the augmented network and Table 7-21to Table 7-26 show the variation in the W_S and the W_P values with different types of sensor and process faults (only a portion of the results are reproduced here, see Tables 5-23 to 5-28 for the complete results).

Table 7-13: Instantiation Table with the Addition of Redundant Nodes

	Sensors						**Processes**				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
1	*NI*	*NoI*					*PoI*				
2	*NI*		*NoI*				*PoI*	*PoI*			
3	*NI*			*NoI*			*PoI*	*PoI*	*PoI*		
4	*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*	
5	*NI*					*NoI*	*PoI*	*PoI*	*PoI*	*PoI*	*PoI*
6		*NI*	*NoI*					*PoI*			
7		*NI*		*NoI*				*PoI*	*PoI*		
8		*NI*			*NoI*			*PoI*	*PoI*	*PoI*	
9		*NI*				*NoI*		*PoI*	*PoI*	*PoI*	*PoI*
10			*NI*	*NoI*					*PoI*		
11			*NI*		*NoI*				*PoI*	*PoI*	
12			*NI*			*NoI*			*PoI*	*PoI*	*PoI*
13				*NI*	*NoI*					*PoI*	
14				*NI*		*NoI*				*PoI*	*PoI*
15					*NI*	*NoI*					*PoI*

Table 7-14: Change in W_S and W_P Values with a Faulty Sensor and the Addition of Redundant Nodes

	Sensors						**Processes**				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
0	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5	0.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	0.8	**0.0**	0.8	0.8	0.8	0.8	1.000	0.5	0.6666	0.75	1.000



DC = PWM Duty Cycle
FR = PWM Frequency
V = Total Motor Terminal Voltage
I = Total Current dran by the motor
τ = Motor Torque
S = Motor Speed
N = Acoustic Noise

Figure 7-6: Simplified Bayesian Network of the Testbed

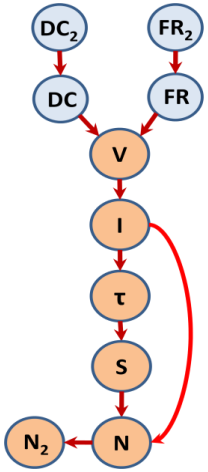


Figure 7-7: Network augmented with redundant sensors

Table 7-15: Instantiation Table for the Testbed Bayesian Network

	Sensors							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
1														
2														
3														
4														
5														
6														
7														
8														
9														
10														
11														
12														
13														
14														
15														

Nodes Instantiated **Node of Interest** **Process of Interest**

Table 7-16: Variation in W_S and W_P Values when all Sensors/Processes are Operating Correctly

	Sensors							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

Table 7-17: Variation in W_S and W_P Values with a Single Sensor Fault

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	0.800	0.800	0.800	0.714	0.800	**0.000**	0.800	0.800	0.800	0.750	0.667	0.800	0.500	0.800

Table 7-18: Variation in W_S and W_P Values with Multiple Sensor Faults

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	0.600	0.600	**0.000**	0.667	0.600	**0.000**	0.600	0.600	0.600	0.375	0.444	0.600	0.375	0.600

Table 7-19: Variation in W_S and W_P Values with a Process Fault

	Nodes							Processes						
Step	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15	0.200	0.200	0.200	0.667	0.600	0.600	0.600	0.200	0.200	**0.000**	0.333	0.600	0.500	0.600

Table 7-20 : Instantiation Table for the Testbed with Redundant Nodes

	Nodes										Processes									
Step	**DC_2**	**DC**	**FR_2**	**FR**	**V**	**I**	**T**	**S**	**N**	**N_2**	**DC_2-DC**	**FR_2-FR**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**	**N-N_2**
1																				
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				
11																				
12																				
13																				
14																				
15																				
16																				
17																				
18																				
19																				
20																				
21																				
22																				
23																				
24																				
25																				
26																				
27																				
28																				
29																				

Node instantiated as evidence Node of interest Process of Interest

Table 7-21 : Variation in W_S and W_P Values with No Sensor or Process Faults

Step	**DC_2**	**FR_2**	**DC**	**FR**	**V**	**I**	**T**	**S**	**N**	**N2**	**DC_2-DC**	**FR_2-FR**	**DC-V**	**FR-V**	**V-I**	**I-T**	**I-N**	**T-S**	**S-N**	**N-N_2**
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
29	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

Table 7-22: Variation in W_S and W_P Values with a Single Faulty Sensor

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
29	0.857	0.857	0.857	0.857	**0.000**	0.909	0.857	0.857	0.857	0.857	1.000	1.000	0.833	0.833	0.667	0.800	0.833	0.824	0.833	1.000

Table 7-23: Variation in W_S and W_P Values with Multiple Faulty Sensors

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
29	0.714	0.714	0.714	0.714	**0.000**	0.636	0.714	**0.000**	0.714	0.714	1.000	1.000	0.667	0.667	0.533	0.550	0.667	0.471	0.667	1.000

Table 7-24: Variation in W_S and W_P Values with a Single Process Fault

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
29	0.286	0.286	0.286	0.286	0.286	0.636	0.571	0.571	0.571	0.429	1.000	1.000	0.167	0.167	**0.000**	0.350	0.417	0.412	0.417	1.000

Table 7-25: Variation in W_S and W_P Values with Multiple Process Faults

Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
29	0.143	0.143	0.143	0.143	0.714	0.545	0.571	0.429	0.571	0.429	1.000	1.000	**0.000**	**0.000**	0.333	0.450	0.417	0.471	0.417	1.000

Table 7-26: Variation in W_S and W_P Values with a Faulty Sensor Corresponding to a Leaf Node

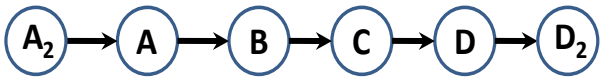
Step	DC_2	FR_2	DC	FR	V	I	T	S	N	N2	DC_2-DC	FR_2-FR	DC-V	FR-V	V-I	I-T	I-N	T-S	S-N	N-N_2
0	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000	1.000	0.500	0.500	0.500	0.500	0.500	0.500	0.500	1.000
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
29	0.857	0.857	0.857	0.857	0.857	0.727	0.857	0.857	**0.000**	0.857	1.000	1.000	0.833	0.833	0.800	0.700	0.500	0.647	0.500	1.000

7.4.2. Factors Affecting the SPFDI Algorithm

Chapter 5 also investigated the effects of four factors viz. the network structure (Section 5.7.1), size of the instantiation table (Section 5.7.2), sensor characteristics (Section 5.7.3), and the effect of node discretization on the choice of thresholds used for comparison (Section 5.7.4) on the effectiveness of the SPFDI algorithm.

Effect of network structure: The network structure has a strong influence on the composition of the instantiation table, which in turn affects the effectiveness of the SPFDI algorithm. A sensor/process may be inadvertently flagged as being faulty by virtue of its column constituting a subset of other columns in the instantiation table, associated with sensors or processes where real faults have occurred. This is referred to as a 'false alarm'. The occurrence of false alarms may be reduced via careful construction of the instantiation table by the domain expert to prevent subsets from being formed as much as possible. However, situations may still exist where it may not be possible to eliminate the potential for false alarms completely. In such cases, it would be desirable to calculate beforehand the odds that a particular sensor/process flagged as faulty by the SPFDI algorithm is indeed faulty. This is vital if there is a possibility that a key sensor/process may end up being flagged as faulty often, despite no real faults occurring.

With redundant nodes, the subset issue and false alarms for single sensor/process faults are largely alleviated as the columns in the instantiation table are made distinct. When multiple faults occur in different combinations of sensors/processes, Section 5.7.1 described a procedure to determine the different subsets that may be formed in the instantiation table for combinations of columns corresponding to the sensors/processes where real faults have occurred. It was observed that not all combinations of real faults (supersets) resulted in the formation of subsets. An example is shown in Figure 7-8 for combinations of two faulty elements (see Table 5-31 and 5-32 for more detailed results). Intuitively, it can be expected that as the number of faulty elements increase, the number of false alarms would also increase (due to a greater number of unreliable versus reliable sources of information, except for the degenerate case when all elements are faulty).



	Sensors						Processes				
Step	**A_2**	**A**	**B**	**C**	**D**	**D_2**	**A_2->A**	**A->B**	**B->C**	**C->D**	**D->D_2**
1	*NI*	*NoI*					*PoI*				
2	*NI*		*NoI*				*PoI*	*PoI*			
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
15					*NI*	*NoI*					*PoI*

Example of All Possible Combinations of Two Faults

1	*B->C*	*C->D*	**13**	*C*	*A->B*	**25**	*A*	D_2
2	*A->B*	*C->D*	**14**	*C*	D_2	**26**	*A*	*D*
3	*A->B*	*B->C*	**15**	*C*	*D*	**27**	*A*	*C*
4	D_2	*C->D*	**16**	*B*	*C->D*	**28**	*A*	*B*
5	D_2	*B->C*	**17**	*B*	*B->C*	**29**	A_2	*C->D*
6	D_2	*A->B*	**18**	*B*	*A->B*	**30**	A_2	*B->C*
7	*D*	*C->D*	**19**	*B*	D_2	**31**	A_2	*A->B*
8	*D*	*B->C*	**20**	*B*	*D*	**32**	A_2	D_2
9	*D*	*A->B*	**21**	*B*	*C*	**33**	A_2	*D*
10	*D*	D_2	**22**	*A*	*C->D*	**34**	A_2	*C*
11	*C*	*C->D*	**23**	*A*	*B->C*	**35**	A_2	*B*
12	*C*	*B->C*	**24**	*A*	*A->B*	**36**	A_2	*A*

Combinations of Faults for which Subsets are formed
(F_i= Faults, SS=subset)

	$F_1 \cup F_2$		**SS**
1	*B->C*	*C->D*	*C*
2	*A->B*	*B->C*	*B*
3	D_2	*C->D*	*D*
4	*D*	*C->D*	D_2
5	*D*	D_2	*C->D*
6	*C*	*C->D*	*B->C*
7	*C*	*B->C*	*C->D*
8	*B*	*B->C*	*A->B*
9	*B*	*A->B*	*B->C*
10	*A*	*A->B*	A_2
11	A_2	*A->B*	*A*
12	A_2	*A*	*A->B*

Figure 7-8: Effect of Network Structure on the SPFDI Algorithm

Table 7-27: Probability of False Alarms

No. of Faults	**Total Combinations**	**Supersets (combinations of real faults that form subsets)**	***P* (False Alarms)**
0	1	0	0.00
1	9	0	0.00
2	36	12	0.33
3	84	60	0.71
4	126	110	0.87
5	126	119	0.94
6	84	79	0.94
7	36	34	0.94
8	9	9	1.00
9	1	0	0.00

This was verified by the results shown in Table 7-27. It can be seen that the SPFDI algorithm is very effective when there are few sensor/process faults (excluding the cases where all or none are faulty). Once a network structure has been designed for a system, and its associated instantiation table has been created, values such as those shownin Table 7-27 may be generated. If it is found that the proportion of false alarms is unusually high, the domain expert may revise the composition of the instantiation table to reduce their

occurrence. With multiple sensor and/or process faults, the probability of each individual element being actually faulty and the probability of a false alarm is calculated as follows:

$$P(Z)_{AF} = \frac{n_{Z_{AF}}}{n_{Z_{AF}} + n_{Z_{FA}}} \text{ and } P(Z)_{FA} = 1 - P(Z)_{AF} \qquad \text{Eq. (7-1)}$$

where Z represents a specific sensor or process, $n_{Z_{AF}}$ is the number of times Z is classified by the SPFDI algorithm as being faulty due to an actual fault, and $n_{Z_{FA}}$ is the number of times Z classified as faulty due to a false alarm for Z, $P(Z)_{AF}$ is the probability that Z is actually faulty and $P(Z)_{FA}$ is the probability of a false alarm being raised for Z (the values for $n_{Z_{AF}}$ and $n_{Z_{FA}}$ can be obtained from tables like Table 5-32). The probability of actual faults and false alarms when two simultaneous faults occur are shown below (shaded cells denote elements flagged as faulty by the SPFDI algorithm).

Table 7-28 : Probability of Actual Faults and False Alarms for Combinations of Two Faulty Elements

	Actual Faults		Subset
1	*B->C*	*C->D*	*C*
2	*C*	*C->D*	*B->C*
3	*C*	*B->C*	*C->D*
4	*A->B*	*B->C*	*B*
5	*B*	*B->C*	*A->B*
6	*B*	*A->B*	*B->C*

	Actual Faults		Subset
7	D_2	*C->D*	*D*
8	*D*	*C->D*	D_2
9	*D*	D_2	*C->D*
10	*A*	*A->B*	A_2
11	A_2	*A->B*	*A*
12	A_2	*A*	*A->B*

$P(A_2)_{AF}$	$P(A)_{AF}$	$P(B)_{AF}$	$P(C)_{AF}$	$P(D)_{AF}$	$P(D_2)_{AF}$	$P(A->B)_{AF}$	$P(B->C)_{AF}$	$P(C->D)_{AF}$
			0.6667				0.6667	0.6667
		0.6667				0.6667	0.6667	
				0.6667	0.6667			0.6667
0.6667	0.6667					0.6667		

$P(A_2)_{FA}$	$P(A)_{FA}$	$P(B)_{FA}$	$P(C)_{FA}$	$P(D)_{FA}$	$P(D_2)_{FA}$	$P(A->B)_{FA}$	$P(B->C)_{FA}$	$P(C->D)_{FA}$
			0.3333				0.3333	0.3333
		0.3333				0.3333	0.3333	
				0.3333	0.3333			0.3333
0.3333	0.3333					0.3333		

Since such values can be generated beforehand, if a particular combination of faults occurs while the system is in operation, then based on the output of the SPFDI algorithm and such charts, the system operator can make an informed decision about the fault status of a particular group of sensors or processes.

Effect of size of the instantiation table: The effect of the size of the instantiation table was discussed in Section 5.7.2. Although the SPFDI algorithm provides an indication of the health of the different elements at any given instant, numerous instantiations and inferences have to be done even to complete a single validation cycle that essentially deals with only one sample of data. To declare a definite sensor/process as faulty entails analysis of multiple samples, which may require considerable computation time and resources. This issue may be greatly alleviated if the time taken for each individual validation cycle is reduced. A potential approach to this is to reduce the number of instantiations in the instantiation table, using the procedure shown in Figure 7-9. However, the tradeoff in terms of speed of execution and the potential for false alarms, between using a full instantiation table and a reduced instantiation table needs further investigation.

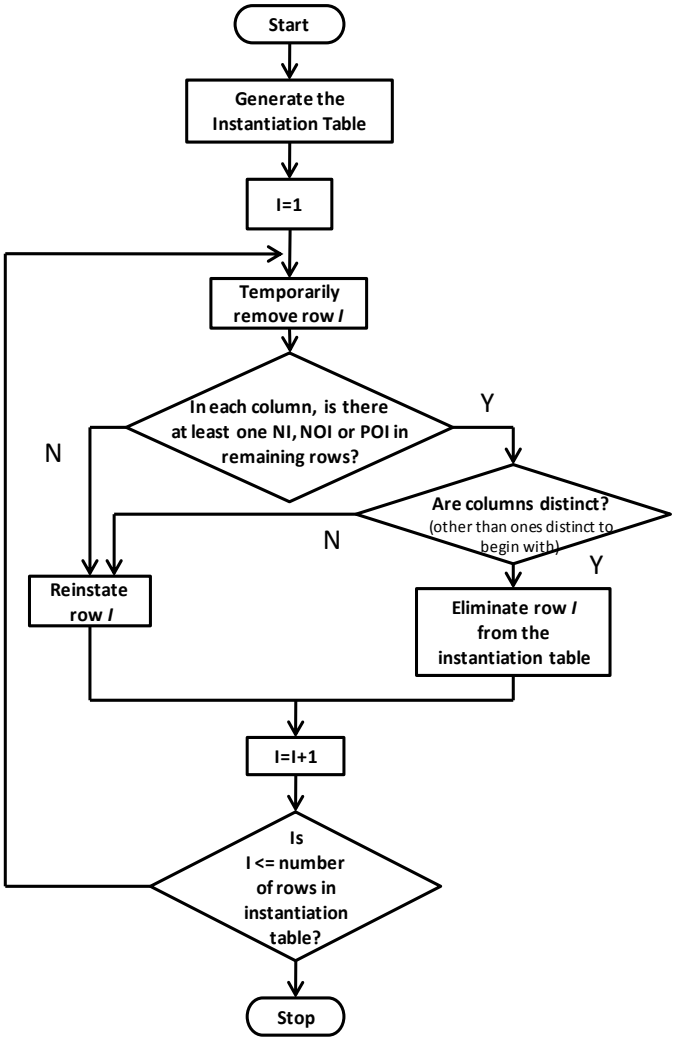


Figure 7-9: Pruning the Instantiation Table

Effect of sensor characteristics: Sensor characteristics like accuracy, resolution, etc. can also play an important role in determining the overall efficacy of the SPFDI algorithm. These factors were explored in Section 5.7.3. The quality of sensors corresponding to the

NI nodes determines which state of these nodes is set as evidence, which in turn plays a role in the inferred value of the *NoI*. Similarly the quality of sensors corresponding to the *NoI* influences the results of comparison of the inferred and the indicated values (when a decision is made if the two values concur, in order to alter the belief values for a particular sensor/process). It is evident that when multiple choices in sensors are available for a specific measurand, characteristics like accuracy, resolution, etc. of the chosen sensor must be commensurate with the application requirements to ensure that the results of the SPFDI algorithm are not disproportionately affected by them (for instance, choosing a sensor with low accuracy may result in wrong node states being instantiated or compared during the execution of the SPFDI algorithm, resulting in more false alarms, while in reality, the sensor and the system may be perfectly healthy)

Effect of sensor characteristics: Section 5.7.4 explored the effect of the level of discretization of the nodes on the choice of thresholds that may be utilized for the SPFDI algorithm. In the algorithm, there is a need to define specific thresholds within which a sensor reading would be deemed acceptable, in order to determine which state of the *NI* or the *NoI* node, to associate with the observed reading. The acceptable limits of variation in values (dictated by application requirements) are used to determine the number of states and bin sizes into which each random variable is discretized. It was observed that a smaller number of states/larger bin sizes allowed the use of bin limits corresponding to the most probable state as the threshold limits for a node; with good results from the SPFDI algorithm in successfully detecting and isolation single as well as multiple sensor or process faults. However, with a greater granularity of discretization, the use of the most probable state and its bin limits as the threshold criteria may result in false alarms. Hence, the use of alternative methods of setting thresholds such as using the expected values of the posterior distribution of a node in conjunction with its variation/standard deviation, using the expected value in conjunction with the bin sizes used to discretize the node states, specifying a certain number of bins centered around the expected value, etc. were suggested. Each of these alternatives would result in thresholds with varying levels of restriction and may be chosen accordingly based on the task being addressed.

7.4.3. Using the Results of the SPFDI Algorithm

To represent the system status at any instant as accurately as possible the initial model parameter values must be regularly updated with fresh data. The objective is to update θ_{ijk}^{T} for the CPTs of all nodes at a sampling instant T (where θ_{ijk} represents the values in the CPT for any node X_i with k states and j configurations of parents PA_i,) to new values θ_{ijk}^{T+1}, when new sensor data $D_{T+1} = \{x_1, x_2....x_N\}$, is obtained. This is known as 'learning' the model parameters. In doing so, it must be ensured that:

a) The data used is reliable enough, to avoid corrupting the existing information, and

b) The process of updating is quick enough to accurately represent the prevalent system conditions, without the need to wait for a large number of data samples.

Section 5.8 proposed an adaptation of the Voting EM algorithm (with adaptive learning rates) developed by [Cohen, et. al, 2001] using the following rule for updating the network parameters, when all the nodes are observable:

$$\theta_{ijk}^{T+1} = \begin{cases} \eta + (1-\eta)\theta_{ijk}^{T} & \text{for } x_i = x_i^k \text{ in } D_{T+1} \text{ i.e. P}(x_i^k | D_{T+1}) = 1 \text{ and } \text{P}(pa_i^j | D_{T+1}) = 1 \\ (1-\eta)\theta_{ijk}^{T} & \text{for } x_i \neq x_i^k \text{ in } D_{T+1} \text{ i.e. P}(x_i^k | D_{T+1}) = 0 \text{ and } \text{P}(pa_i^j | D_{T+1}) = 1 \\ \theta_{ijk}^{T} & \text{otherwise} \end{cases} \quad \text{Eq. (7-2)}$$

where $0 \leq \eta \leq 1$. As $\eta \rightarrow 0$, the value of past data is weighted significantly and model parameters remain practically unchanged. Conversely, as $\eta \rightarrow 1$, newly available data is assigned a higher importance. Faster convergence to true probability values is obtained with values of $\eta \rightarrow 1$, and vice-versa. The choice of an appropriate η value is thus crucial.

The initial values of η for different nodes may be determined by the user based on the number of samples (this value is application-dependent) that have been analyzed previously. For instance, for a new, relatively untested system, it would be desirable to have a high initial η value for fast convergence to estimate the parameters quickly. On the other hand, if the system has been operational for a while, it is likely that significant data has already been analyzed to estimate the CPT parameters. In such a case, a low starting value of η might be sufficient.

To decide when and by how much to modify the learning rate, the use of the W_S and W_P values obtained from the SPFDI algorithm was proposed both as the decision criterion to modify η and also to determine the magnitude of change. If all the sensors and processes are operating correctly (W_S and W_P values ≈1 or above user-defined thresholds), η can be modified as follows. For a particular combination of node value X_i and its parents in the configuration PA_j, if $\eta^L_{PA_jX_i}$ and $\eta^H_{PA_jX_i}$ are the lowest and highest values of η respectively, *NS* are the number of data samples that have been previously analyzed, the new value of $\eta_{PAj,Xi}$ for the next learning cycle can be calculated as:

$$\eta_{PA_jX_i} = \eta^L_{PA_jX_i} + \frac{\eta^H_{PA_jX_i} - \eta^L_{PA_jX_i}}{1 + NS} \qquad \text{Eq.(7-3)}$$

As *NS* increases, η decreases from $\eta^H_{PA_jX_i}$ and attaining a value of $\eta^L_{PA_jX_i}$ after numerous samples. Practically, the learning rate decreases but remains a finite value.

If, a sensor corresponding to a node X_i is identified as being potentially faulty at the end of the fault isolation cycle preceding the latest data sample (indicated by its $W_S \approx 0$ or lesser than a user-defined threshold), then

a) The $\eta_{PAj,Xi}$ for all the columns in the CPT of X_i is set to a value of zero.

b) As the sensor for X_i influences the parent configurations PA_r for all its child nodes Y_k, the learning rates for all those nodes, $\eta_{PAr,Yk,}$, are also set to zero (to prevent corruption of the existing CPT parameters of X_i, if the sensor is actually faulty)

If it is found that all the sensors are operating correctly but there is a process fault (one or more W_P values are ≈0 or below certain user-defined thresholds), $\eta_{PAj,Xi}$ for that process needs to be increased to enable quick updating of the CPT parameters to represent the variation in system dynamics. If PA_j represent the particular configuration of the parents P_1, P_2, …P_m of the node X_i, and there is a fault in the k^{th} process P_k -> X_i, the new learning rate may be calculated as follows:

$$\eta^{new}_{PA_jX_i} = \eta^{current}_{PA_jX_i} + \left(1 - W_{P_{avg}}\right)\left(\eta^H_{PA_jX_i} - \eta^{current}_{PA_jX_i}\right) \qquad \text{Eq.(7-4)}$$

where $W_{P_{avg}}$ is the average of the W_P values obtained by considering all the processes, viz. P_1-> X_i, P_2-> X_i,.. P_m-> X_i that terminate at X_i. It can be seen that the new learning

rate is determined by the system condition; the faultier it is (low W_P values for multiple processes), the higher is the learning rate. If all the processes that terminate at X_i and all the sensors are operating normally, $\eta_{PAj,Xi}$ remains unchanged. In the extreme scenario when all the processes are faulty ($W_{P_{avg}} \approx 0$), then $\eta_{PAj,Xi}$ is set to its highest value. For any other intermediate condition ($0 < W_{P_{avg}} \leq 1$), $\eta_{PAj,Xi}$ is increased from its present value.

7.5. APPLICATIONS

The SPFDI algorithm does not place any restrictions on the types of systems or sensors it can be applied to nor does it require assumptions to be made regarding the condition of the monitored system or the sensors. The prerequisites for implementing the algorithm are the existence of a Bayesian network representing the functional relationships/conditional independencies among the variables pertinent to the system and the ability to measure every node during operation using actual physical sensors or abstract sensors (like a knowledge base or human opinion). To demonstrate the spectrum of potential application avenues beyond intelligent EMAs, four broad application classes were studied as described below:

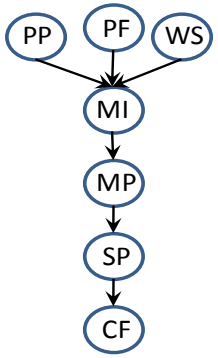
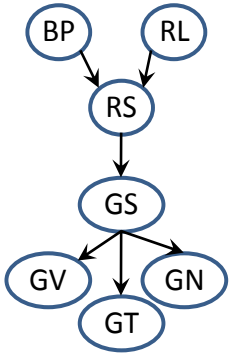
- **Human Safety at Stake** (Chapter 6, Section 6.2.1)**:** In such applications, a significant degradation in the system performance capability (real or perceived due to faulty sensors) may jeopardize the safety of humans interacting with it. If all sensors indicate expected values under the prevailing operating conditions, the operator has assurance in the fact that the system is operating normally. However, if values from one or more sensors deviate from the anticipated values, the decision maker needs to accurately estimate the system capability before making any operational decisions to gauge the risk associated with the decision.
- **System Availability** (Chapter 6, Section 6.2.2)**:** In these applications, the decision maker has to decide if a system in its present condition is capable of performing the tasks required, based on sensor data obtained during a start-up/ performance check process that puts it through a series of selected operational maneuvers representative of its actual duty cycle. This is termed as 'system availability'. The

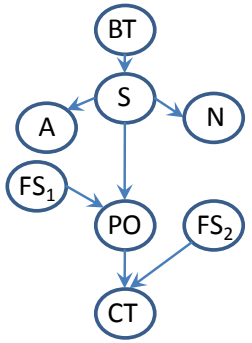
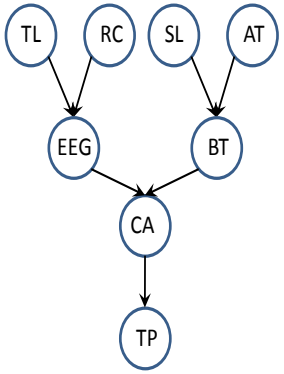
goal is to avoid a costly system shutdown once it is deployed (for example, due to a false alarm) or a catastrophic complete failure if potential sources of single point component faults remain undetected before the system is deployed.

- **High Cost of System Failure** (Chapter 6, Section 6.2.3)**:** This class encompasses applications where the decision maker may temporarily shut a system down for remedial action, due to its perceived inability to accomplish a task or command it to operate well beyond its design limits for an extended time due to its apparent ability to do so. In either case, an erroneous decision resulting from inaccurate estimation of the system performance capability (for instance, due to faulty sensors) may result in a tangible economic cost in terms of lost productivity or in the extreme case, an irreparable damage or complete loss of the system itself.
- **Performance Level Maximization** (Chapter 6, Section 6.2.4)**:** The objective in this class of applications is to enable the decision maker to decide on the optimum allocation of the available resources in highly reconfigurable systems. The goal is to provide the best possible (maximum) system performance to address task requirements that may lie beyond nominal design/operation specifications or to overcome any significant deterioration in the system performance capability.

Four representative applications were examined for each application class and one example from each class was explored in depth to demonstrate the use of the SPFDI algorithm to as well as the utility of the belief values obtained for the sensors and processes, for higher level decision-making by the system operator (Table 6-1). The unifying characteristic of all the application classes is that a complex system is monitored using multiple heterogeneous sensors with human choices involved in the operational decision making process. Implementing the SPFDI algorithm in all these applications can enable the decision maker to determine the reliability of the sensor values obtained at any given time and judge accurately using the W_S and W_P belief values obtained if any observed unanticipated variations in readings are due to regular system changes or if they are indicative of any potential faults.

Table 7-29: Four Potential Classes of Application Areas for the SPFDI Algorithm

Class	Applications / Section	Example
Human Safety at Stake	• Aircraft subsystems (Section 6.2.1) • Robotic surgery (Section 6.2.1) • **Automobiles** (Section 6.3) *PP* = Pedal position *PF* = Pedal force *WS* = Wheel speed *MI* = Motor current *MP* = Motor position *SP* = Caliper position *CF* = Caliper force	Bayesian network for a brake-by-wire system 
High cost of system failure	• Manufacturing line (Section 6.2.3) • Remotely operated vehicles (Section 6.2.3) • **Wind turbines** (Section 6.5) *BP* = Blade pitch angle *RL* = Wind load *RS* = Rotor speed *GS*= Generator speed *GV* = Gearbox vibration *GT* = Gear temperature *GN* = Gearbox noise	Bayesian network for a wind turbine gearbox 

Class	Applications / Section	Example
System Availability	• Battlefield robots (Section 6.2.2) • Submarines (Section 6.2.2) • **Nuclear power plant** (Section 6.4) *B T* = Bearing temperature *A* = Accelerometer *N* = Acoustic noise *S* = Pump operating speed *PO* = Volumetric pump output *CT* = Core temperature *FS1*, *FS2* = Flow sensors	Bayesian network for a coolant circulation system 
Performance maximization	• Ground combat vehicle (Section 6.2.4) • Manufacturing cell (Section 6.2.4) • **War fighter monitoring** (Section 6.6) *TL* = Threat level *RC* = Rest condition *SL* = Soldier combat load *AT* = Ambient temperature *EEG* = Electroencephalography *BT* = Body temperature *CA* = Cognitive ability *TP* = Task performance	Bayesian network Soldier task performance 

7.6. CONCLUSIONS AND RECOMMENDATIONS FOR FUTURE RESEARCH

The Sensor and Process Fault Detection and Isolation (SPFDI) algorithm developed in this work represents a novel approach to simultaneous detection and identification of sensor and process faults using a Bayesian network-based approach and represents significant advancement of existing state-of-the-art approaches. The creation of a detailed Bayesian network for a switched reluctance motor and the specification of design and operational criteria for resource management are the other major research contributions from this work. Although many associated aspects were explored while developing the SPFDI algorithm, the work presented in this report is still nascent and can be considered as the foundation for future research that can further enhance the effectiveness as well as application of the SPFDI algorithm in numerous applications. The following section presents five potential avenues where in-depth future research can vastly help enhance the work presented in this report and may be organized as:

- Short-term work i.e. tasks that follow directly from the present work (Items 1, 2, and 3; may be accomplished in a suggested timeframe of 1-3 years).
- Long-term work that would draw from the various research threads at UT-RRG (and externally) to help advance the field of intelligent mechanical systems (Items 4 and 5; may be accomplished in a suggested timeframe of 3-5 years).

1. Refinement of the SPFDI Algorithm

a. *Improving the construction of the Instantiation Table*: In the present version of the SPPFDI algorithm, only predictive reasoning has been considered in generating the instantiation table. The use of a physically relevant, causal basis for interpreting the relation between the variables provided an intuitive starting point for implementing the algorithm. However, the use of a Bayesian network framework allows the use of predictive as well as diagnostic reasoning by designating the appropriate nodes as evidence and query (Section 2.5). This

flexibility can be exploited to improve the construction of the instantiation table (Section 5.4.2.1) with many potential benefits.

Given the conditional independencies encoded by the Bayesian network some of the instantiations included in the instantiation table can either be completely eliminated or combined into a single step. For instance, this could be achieved by considering all the nodes in the Markov blanket[1] of a specific node as the *NI* in a single step of the fault detection and isolation cycle. Reducing the size of the instantiation table in this manner can significantly improve the speed of execution of the algorithm. In other cases, it may be possible to almost completely eliminate the subset issue (Section 5.5.3.1) by introducing additional instantiations via abductive/diagnostic reasoning (setting descendent nodes as *NI* and considering ancestral nodes as the *NoI*). Both these possibilities need to be explored further.

b. *Exploring the Tradeoff Between Execution Time and False Alarms*: In Section 5.5.4, the use of redundant nodes was proposed to overcome the subset issue. Although this enables additional instantiations to be introduced and reduces the potential for false alarms, it increases the size of the instantiation table. This in turn leads to a longer execution time for an individual iteration of the SPFDI algorithm. For an application with a complex network, this may even render the size of the instantiation table unmanageable to achieve any timing constraints that are imperative for the application.

In Section 5.7.2 a procedure was proposed to reduce the size of the instantiation table and improve the speed of execution of the algorithm. Although a smaller table may lead to a faster algorithm cycle (and possibly meet the conditions required for real-time operation), there may be potential drawbacks in reducing the number of instantiations in this manner. Depending on the threshold

[1] The Markov blanket of a variable/node may be considered as the set of variables/nodes that make it independent of the other variables/nodes. [Pearl, 1988] defines the Markov blanket for any node X in a Bayesian network as the union of three types of neighbors for X: the direct parents of X, the direct successors of X and all the direct parents of X's direct successors

values used to determine the acceptability of an observed deviation in the sensor reading, the larger granularity of change in the W_S and W_P belief values[2] may result in more instances of false alarms for a sensor/process depending on which rows have been eliminated. If a sensor that is critical from the application point of view ends up being flagged as potentially faulty often as a result of such row eliminations, it may be necessary to introduce additional nodes or instantiations to overcome the situation.

Thus, depending on the objectives of the application in which the SPFDI algorithm is intended to be used (for instance, avoiding false alarms in a process) and the available computational resources, the ramifications of using a full instantiation table (the original table augmented with additional instantiations resulting from redundant nodes) versus a reduced instantiation table (obtained by eliminating rows from the original or the full instantiation table) in terms of execution speed, potential for false alarms, etc. need to be explored in greater detail. Based on the specific application, it may be possible to define multiple levels of real-time operation depending on different operating regimes of the system for which different approaches described above may be suitable[3].

c. *Comprehensive Evaluation of the Effectiveness of the SPFDI Algorithm*: The issue of false alarms that may arise from the SPFDI algorithm due to the subset issue was addressed in Section 5.7.1. The false alarms primarily dealt with the situation of Type I errors wherein a non-faulty sensor or process is wrongly classified as being potentially faulty (false positive). The other type of errors, referred to Type II errors, represent the other end of the spectrum where a potential sensor or process fault remains undetected by the SPFDI algorithm (false negative). In the

[2] The incremental amounts by which the W_S and W_P belief values are modified depend on the number of rows a particular sensor or process occurs in the instantiation table.

[3] It must be noted that the term 'false alarm' as used throughout this work is primarily used in the context of sensor-level decisions. The goal of reducing misdiagnosis of faults at the sensor level is to reduce the potential for false alarms from CBM algorithms, which are primarily overall system-level decisions. In the latter context, the term 'false alarms' are used to refer to a wrong estimation of system operational capability, or misestimating the remaining useful life of the system [Hvass and Tesar, 2004][Tesar, 2009]

present work, these errors were not encountered due to the nature of the faults that were artificially introduced into the sensor readings used as well as due to the thresholds that were used used to determine a sensor fault.

In order to fine-tune the algorithm for any application, a detailed analysis of the latter type of errors is essential to prevent any undesirable consequences from occurring as a result of undetected faults once the system has been deployed. To provide an estimate of the possibility of such errors occurring requires extensive testing of the algorithm (via actual experimentation or simulations) for different fault scenarios for all the sensors in the application or processes that occur in the system (for instance, by defining varying degrees of faults such as mild, medium, severe for every sensor/process and exploring the effects of each fault level on the other variables individually as well as the overall system behavior), exploring the use of different types of thresholds in conjunction with the different fault scenarios (for instance, investigate the variation in the frequency of occurrence of type II errors as different thresholds are used or as the thresholds become more lenient or restrictive).

d. *Improving the Procedure for Updating of the Model Parameters*: In Section 5.8, the discussion was centered on the use of the W_S and W_P belief values obtained from the algorithm in updating the model parameters (using Eq. (5-6) and Eq. (5-7)). In the procedure described it was proposed that if a specific sensor X was found to be faulty, then, in order to avoid corruption of the existing decision surfaces by faulty data, the learning rate for X and all its child nodes were to be set to zero. However, a sensor may be mistakenly classified as being faulty due to the subset issue. Ignoring the data from the sensor in this case may result in the loss of valuable information which may be used to update the existing decision surfaces (for instance, utilizing the data could potentially assist the higher level CBM algorithm detect any degradation in the system performance more quickly). Thus, there is a need to explore some mechanism that can, instead of ignoring the data from a misclassified faulty sensor completely, take into account the known

probability of false alarms for that specific sensor (Section 5.7.1) arising from the subset issue, and modify the updating equations accordingly .

2. Algorithm Implementation

a. *Understanding the Implications of Implementing the SPFDI Algorithm on Different Platforms*: The present testbed configuration consists of a host PC connected to a cRIO-9112 chassis with a FPGA and a NI-9014 Real Time Controller (RTC) (see Appendix B for details). With this hardware configuration there are three execution platforms available to run different sections of the Actuator Management and Operation Software (AMOS) code for motor control, data acquisition (see Appendix C). In general, processes that need to be deterministic and quick such as data acquisition and control are programmed on the FPGA. Operations that are relatively less time critical such as data processing may be run on the RTC. The remaining operations are run on the PC like transferring data between the different AMOS modules, the graphical user interface for AMOS, Bayesian network inferencing algorithms, etc.

The present version of the SPFDI algorithm was implemented on a standalone PC and in a non-synchronous mode (considering the execution time for the algorithm relative to the motor operational duty cycle, buffered data was used). Given the ample memory storage and processor capability, the algorithm was able to generate good results relatively quickly (with an execution time averaging less than 2 seconds for the network demonstrated in Chapter 5). However, in certain applications, the flexibility to partition the AMOS execution code to different processors may not be possible. As indicated in Chapter 4 (Sections 4.3.9 and 4.4.5) the available computational resources (processing power as well as memory) may need to be shared between different operations that are needed for system operation. For instance, in a mobile robot, all the operations ranging from control, data acquisition to navigation and communication, are typically executed on an on-board computer. The present

version of the code for the Bayesian network inferencing algorithms as well as the overall logic for the SPFDI algorithm needs to be optimized further (for handling large networks with highly discretized nodes, generating a large number of samples quickly for approximate inferencing algorithms, etc.) to enable their execution on such embedded systems concurrently with other programs.

b. *Extension of the SPFDI Algorithm to Cover Transient Conditions*: Based on the execution time required for the present version of the code for the SPFDI algorithm, a necessity in implementing it is that all the data samples used be collected after the values have stabilized since individual measured values are compared with the inferred values at every sampling instant. In other words, the data utilized for the algorithm is buffered during the actual operation of the test bed once the steady state values for all the variables have been attained (i.e. within acceptable limits of variation) .

The algorithm needs to be extended to cover transient conditions as well where all the variables may be varying rapidly and some of them may never attain a true steady state value. In such a situation, the W_S and W_P values may fluctuate unpredictably (for instance, a sensor X may be flagged as faulty based on the sample at time T while it may be identified as being non-faulty based on the sample at time $T+1$). While this may be as simple as improving the execution time of the algorithm to such an extent that analyzing the data samples at every sampling instant becomes possible so that each sample can be considered to be a quasi-steady state. To incorporate the transient nature of variables, it may be necessary to explore the use of dynamic Bayesian networks.

c. *Providing a Better User Interface for the SPFDI Algorithm:* Although the W_S and W_P belief values provided by the present version of the algorithm provide a good indication of the potential sources of unanticipated deviations in sensor readings, the comprehension of those values can still present a challenge to the average end user. Especially for a complex system represented by a network with a large number of nodes/sensors and links/processes, the sheer magnitude of the results

obtained from the algorithm can be overwhelming. Given the emphasis on the use of visual representation of decision surfaces, use of Bayesian networks for modeling, etc.; all with a view to improve human understanding of the system behavior for improved decision making, another future avenue to be explored is the design and use of a suitable, easy to comprehend graphical user interface for displaying the results of the SPFDI algorithm (for instance, generating graphs for the learning rate η or the W_S and W_P values for key decision variables for different types of sensor/process faults, etc.). The resultant interface could be integrated to work in concert with the other graphical display components of AMOS as described in [Ashok and Tesar, 2007] and [Yun and Tesar, 2006].

3. Definition and Implementation of Additional Resource Management Criteria

A number of criteria (design and operational) were proposed in Chapter 4 as a basis for a criteria-based resource management framework to determine the optimal set of sensors to use at any given time. However, only the most basic criteria have been explored in this work. An exhaustive study of additional general criteria that could be applied across diverse application domains as well as more application-specific ones needs to be pursued in the future.

For instance, a list of sensor-specific design criteria may be defined for each of the ten measurands recommended for intelligent EMAs (see Appendix A). These criteria could deal with the physical form of sensors (volume/dimensions, weight, individual modules/integrated package, etc.), performance specifications (resolution, accuracy, linearity, bandwidth, analog/digital output, voltage/current output, etc.), and other miscellaneous aspects (technological principle on which the sensing element operates, materials used in the sensor construction, hard-wired or wireless, contact/non-contact, cost, degree of hardware redundancy needed based on the application, self-generating or requiring an external power source, communication protocol required and the associated data transfer rates, etc.). The creation of decision surfaces for characterizing sensor selection and

performance/utilization (like those used at the EMA level) is also a distinct possibility. The eventual goal of such an effort would be to create an interactive configuration management software tool (populated and regularly updated with a catalog of specifications for all the sensors of interest) that could be used to structure a sensor system for new applications or expand an existing one. Such a tool can enable a system designer to decide suitable sensors that can be integrated in an application, assist the domain expert to develop more usable Bayesian networks, and finally help the system operator make relevant decisions for prioritizing sensor, accommodating failed sensors, etc.

Additionally, the suitability of the various operational criteria suggested in this work, for real-time operation also needs to be explored further. As an initial step, some criteria may be implemented as part of the AMOS sensor module (Appendix C) and tested on the test set up at UT-RRG (Appendix B).

4. Exploring the Use of the SPFDI Algorithm in Additional Application Areas

The use of the SPFDI algorithm was demonstrated for an EMA in Chapter 5 and proposed for four different classes of applications in Chapter 6. This provides a starting point to further explore its use in a broader gamut of applications. Communication needs to be established with experts in various domains to refine the understanding and the intricacies of applying the algorithm to those domains (for instance, to develop accurate Bayesian networks, developing instantiation tables for unique monitoring of every sensor/process without encountering subset issues, appropriate thresholds to be used, etc).

In all the applications discussed in this work, the sensors/processes considered for demonstrating the algorithm pertained to what could essentially be considered as a basic, standardized building block of higher level system of systems(such as an EMA, a brake-by-wire unit, a coolant circulating system, a gearbox and a soldier). In each application, these fundamental units work in concert with other similar units as well as other independent components which

may have their own set of sensors/processes. For instance, in a typical industrial robot, six or more independent EMAs may be used to actuate the joints, each with its own custom design and set of sensors. Other large systems like an automobile or an aircraft are also similarly composed of numerous lower level systems, each with a diverse array of sub-components and sensors. For effective operation of the complete system, it is imperative to ensure that each of the lower level systems is performing to its optimum potential. As a system increases in its complexity, it may become necessary to implement the SPFDI algorithm for each low level system; generating separate instantiation tables for each. The results from the individual SPFDI algorithms may then be analyzed individualy or as part of an overall higher level CBM algorithm to determine the effect of any potential sensor/process faults that may have occurred in any of the lower level systems on the health or operational capability of the overall system. It may become necessary to investigate methods other than those explored in this work for reducing the potential for false alarms even as the complexity of the networks increases.

Another possibility is to extrapolate the philosophy of the SPFDI algorithm beyond the constituent sensors/processes that occur within each of the aforementioned standardized building blocks (say, for example, distributed EMAs) by considering the entire basic unit as directly analogous to ‘sensors’ in the context of this work and the interaction between these units as the ‘processes’ and using these definitions to generate the instantiation table for the complete system-of-systems. Such an approach could be used directly for fault detection and isolation in applications such as multi degree of freedom robotic workcells with multiple EMAs working in concert.

5. Detailed development of AMOS modules

While software code was developed during this work to address different goals like testbed operation and control, data acquisition, Bayesian inferencing,

implementing the SPFDI algorithm, etc. it was developed on an as-needed basis and can be considered as a starting point for a more structured approach to develop comprehensive as well as refined code that can be seamlessly integrated into the AMOS libraries. To enable AMOS to address the requirements of the widest possible range of applications and depending on the level of warranted operator intervention in the application being considered, there is a need to continuously refine the existing software development goals as well as define new objectives that can address more stringent real-time operating constraints, provide better user and application programming interfaces, determine the depth of detail required for the different modules as well as the level of code abstraction and encapsulation, etc. With different AMOS modules in varying stages of development presently, it is expected that the software could soon be deployed in numerous applications in the near term future.

Appendix A. Electromechanical Actuator Sensing Requirements

To determine the minimum set of sensors that can be utilized across a broad range of applications, a detailed analysis of factors affecting the behavior of various EMA components (and in turn, the overall EMA performance) was carried out and a ten-sensor architecture (with measurands such as position, velocity, acceleration, torque, temperature, noise, vibration, current, voltage, magnetic field) was proposed in [Krishnamoorthy and Tesar, 2005] to provide complete awareness of variations in the different EMA characteristics, and summarized in the table below.

Table A-1: Summary of sensing requirements for intelligent EMAs

Note: The following table is reproduced partially from [Krishnamoorthy and Tesar, 2005]

Component	Objective	Purpose/ Phenomena Monitored	Recommended Sensors	Description / Importance
Geartrain	Condition Monitoring	Motion Nonlinearities	Position, temperature sensors	Compensation for lost motion due to backlash, deformation, hysteresis etc.
		Frictional Nonlinearities	Position, velocity, torque sensors	To provide friction compensation capabilities in actuator control algorithms, monitor gear sliding losses
		Power loss, acoustic emission and vibration monitoring	Position, torque, temperature sensors, accelerometer, microphone	Monitor transmission error, vibrations and noise, incipient defects, wear, etc.
Bearings	Condition Monitoring	Fluting	Current, voltage sensors	To prevent bearing damage due to shaft currents, monitor lubricant viscosity, detect incipient bearing damage
		Wear/misalignment	Accelerometer, microphone	
		Losses	Temperature sensors	

Table A-1: Summary of sensing requirements for intelligent EMAs (contd.)

Component	Objective	Purpose/ Phenomena Monitored			Recommended Sensors	Description / Importance
Prime Mover	Control	Commutation			Hall effect position sensor/encoder	Determine the sequence of energizing stator phases (based on rotor position)
		Feedback			Current, position, velocity, acceleration, torque	Feedback for motor control loops
	Condition Monitoring	Electrical Losses	*Copper/ Ohmic Loss*		Current sensor	Determine the overall prime mover efficiency as well as to monitor the prime mover response to torque demands
			Iron Loss	*Eddy Current Loss*	Magnetic field sensor	
				Hysteresis Loss	Voltage, magnetic field sensors	
		Mechanical Losses	*Friction*		Velocity sensor	
			Windage		Velocity sensor	
		Thermal Nonlinearities			Temperature sensor	Monitor temperature
		Electromagnetic Nonlinearities			Current, magnetic field, temperature sensors	Infer flux leakage, bearing damage etc.
		Saturation- Current/Voltage/Magnetic			Current, voltage, magnetic field sensors	Determine motor torque and speed
		Torque Ripple			Torque, current, position sensors	'Quality' of output torque; monitor vibrations and speed variations
		Internal Noise and Vibration			Accelerometer, Microphone	To monitor vibrations, bearings/rotor imbalance, etc
Actuator Controller	Condition Monitoring	Thermal Losses			Current, voltage, temperature sensors	Monitor switching losses, I^2R losses, temperature rise, voltage, spikes etc.

Appendix B. EMA Software Development Testbed

The test bed has been designed to be modular and easily expandable to accommodate different types of EMAs (with different prime movers such as a switched reluctance motor, brushless/ brushed DC motors, stepper motor, etc. in combination with different geartrains), sensors, loading devices (brakes, load motors), etc. In the present configuration of the test bed (Figure B-2), a four phase Switched Reluctance Motor (SRM) drives a shaft coupled to a hysteresis brake. Voltage and current sensors measure the phase as well as line supply values. An in-line rotary sensor measures the motor torque. Hall-effect sensors are embedded in the motor phases for commutation as well as to determine the shaft speed. A quadrature incremental encoder is also used to calculate shaft position, velocity and acceleration. Vibration is measured using a piezoelectric accelerometer. An acoustic noise sensor is also used. The entire set-up is controlled using a National Instruments (NI) Compact Reconfigurable Input Output (cRIO) platform. The motor and brake are controlled using signals from a digital output module. The sensor signals are read using the analog or digital input modules. Calibration, hardware/ software signal conditioning of the sensor signals are also performed on the cRIO. The following sections provide a brief description of each component in the test bed.

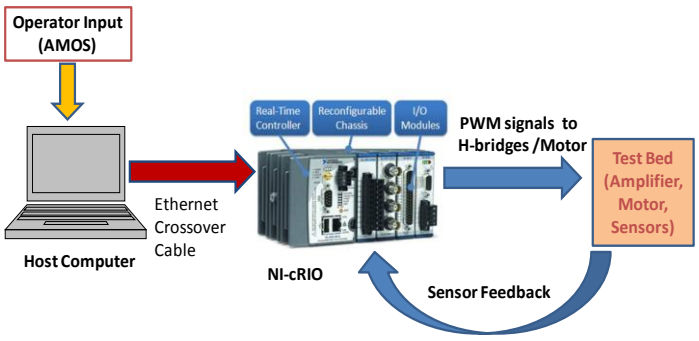


Figure B-1: Test-bed Architecture

B.1 The Compact Reconfigurable Input Output (cRIO) Platform

The cRIO platform is a reconfigurable embedded data acquisition and control system that can be used for applications that require a high performance and reliability.

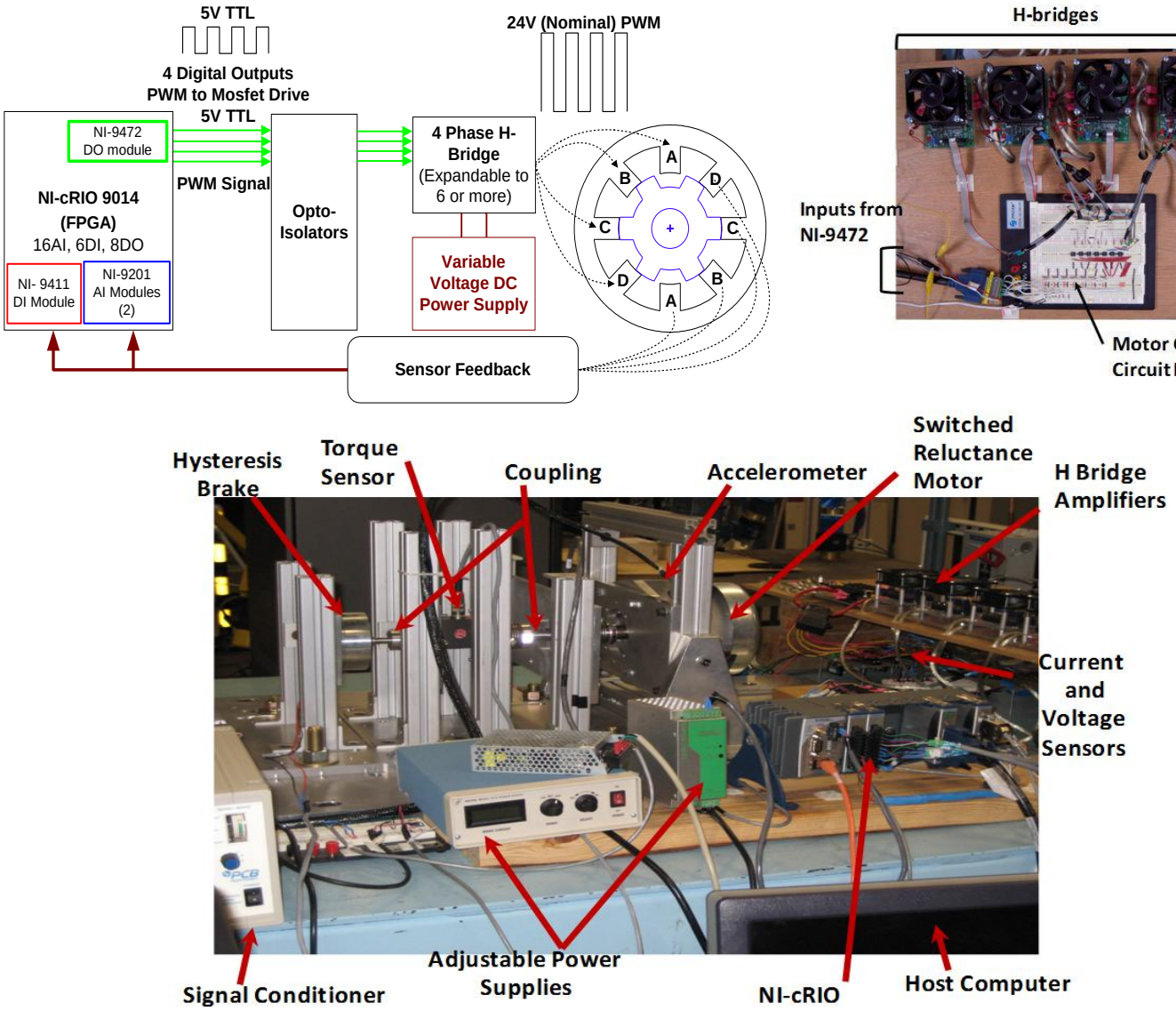


Figure B-2 : Intelligent Actuator Software Development Testbed

The chassis along with the real-time controller (NI-9014) acts as a stand-alone computer. The system has a small footprint, open embedded architecture and is very rugged (-40 to 70°C operating temperature and 50g shock rating) with hot swappable I/O modules [National Instruments, 2009]. It also has network connectivity and a host of sensor measurement capabilities. The C-Series measurement modules installed in the chassis (NI-9201 for analog inputs, NI-9472 for digital outputs, and NI-9411 for digital inputs) have built-in signal conditioning and allow for direct connection to a variety of sensors and I/O signals. The system also has nonvolatile memory for local data storage. The applications are deterministically executed on a Wind River VxWorks real-time operating system [National Instruments, 2009].

The chassis also has a high-performance Field Programmable Gate Array (FPGA) providing dedicated bandwidth for high-speed data processing. It can be re-programmed using simple tools provided in LabView™ software for custom timing, synchronization, control, and processing of analog and digital I/O signals. It consists of logic blocks (with digital components like multipliers, multiplexers, etc.), programmable interconnects, and I/O blocks. The bits are processed in these blocks to create programmatic results. They are connected with programmable interconnects that route signals from one block to the next as well as to the I/O blocks that are connected to the pins on the chip for two-way communication to external circuits (like H-bridge amplifiers) [National Instruments, 2009]. LabView™ generates VHDL code and passes it to a Xilinx compiler, which synthesizes it and routes all synthesized components into a bitfile. This bitfile is then downloaded to the FPGA on the cRIO to perform the desired task. The FPGA is used in lieu of a conventional hardwired motion control board as it accords greater flexibility in designing and implementing different motion control algorithms on demand, as compared to a predetermined set of algorithms provided by the motion controller manufacturer. In a FPGA based system, concurrent execution of parallel code is achievable since the code is implemented as parallel circuits in hardware (greater determinism in execution).

The control set-up consists of a variable voltage DC power supply (the Lambda Genesys programmable DC power supply), H-bridge amplifiers, an opto-isolator bank (to

isolate the low power PC/cRIO side from the high power motor side) and the cRIO system. It is possible to communicate with the cRIO from any computer on a network. User inputs are acquired via the AMOS user interface (Appendix C) and communicated to the cRIO through an ethernet crossover cable. The number of H-bridge units used depends on the number of independent motor phases (four are used in the present test set-up). Each H-bridge unit amplifies a 5 V TTL Pulse Width Modulated (PWM) signal from the cRIO digital output to a 20 V signal that is applied across the motor terminals. With the variable voltage power supply, the voltage supplied to the motor can vary from 0-48 V based on PWM signals from the FPGA.

B.2 Reconfigurable Chassis

The NI cRIO-9112 is an eight-slot reconfigurable embedded chassis. It features a user-programmable Xilinx Virtex-5 FPGA board [National Instruments, 2009]. Other units such as the real-time controller module and the analog and digital I/O modules are plugged into this chassis, which also provides low-level hardware access to all those modules. Additional modules (for specific communication protocols like CAN/serial communication, memory or motion control module for brushed DC motors) can be added to the chassis based on the application.

[National Instruments, 2009]

Figure B-3: cRIO-9112 Chassis

Table B-1: cRIO-9112 Specifications

[National Instruments, 2009]

Manufacturer	National Instruments	
Model No.	NI cRIO-9112	
Parameter	**Value**	**Units**
No. of flip-flops	19200	
No. of 6-input LUTs	19200	
No. of DSP48 slices	32	
Embedded RAM	1152	Kbits
Timebase	40	MHz
Operating Temperature	-40 to 70	^{o}C

B.3 Real-time Controller

The NI-9014 embedded controller features an industrial 400 MHz Freescale MPC5200 real-time processor running LabView^{TM} Real-Time for deterministic tasks like control, data logging, and analysis [National Instruments, 2009].

[National Instruments, 2009]

Figure B-4: NI-9014 Real Time Controller

It is designed for extreme reliability, and low power consumption. The controller is especially suited for functioning for long periods of time in remote applications. The controller features built-in web (http) and file (ftp) servers to stream data from the controller to a PC for further processing, data logging, etc. It is also possible to connect external storage media to the controller for extended on-board data logging. The FPGA board in the cRIO chassis controls the I/O modules and passes data to the controller through a local PCI bus.

Table B-2: Real Time Controller Specifications

[National Instruments, 2009]

Manufacturer	National Instruments	
Model No.	NI-9014	
Parameter	**Value**	**Units**
Supply Voltage	9-35	V
Operating Temperature	-40 to 70	^{o}C
Connectivity	10/100BASE-T Ethernet port, RS232 Serial port	
Memory	2 GB nonvolatile, 128 MB DRAM memory	

B.4 Analog Input Modules

The NI-9201 analog input modules provide high-quality, high-performance measurements for a variety of sensors. In the present test set-up the outputs from current and voltage sensors, torque sensor, accelerometer and acoustic noise sensor are connected to two such modules in the cRIO chassis. Each module includes built-in signal conditioning and integrated connector with screw terminals for flexible and easy wiring of signals. The modules connect directly to FPGA hardware and provide a range of options for sampling rates, synchronization of sensor signals, etc. The modules are hot-swappable during operation and have channel-to-earth ground double-isolation barriers for safety and noise immunity [National Instruments, 2009]. They also have a high common-mode voltage range. A partial list of specifications is provided in Table B-3.

B.5 Digital Input Module

The NI-9411 digital input module is a high-performance digital input-switching module for the cRIO system. It provides industrial logic levels for direct connection to a wide array of industrial switches and transducers [National Instruments, 2009]. In the present test set-up, signals from the Hall-effect sensors embedded in the motor phases (for commutation) are connected to this module. The sensors produce a digital high signal when a rotor pole is in complete alignment with a stator pole, and a digital low otherwise signal otherwise[1]. The digital signals (channels A, B, index) from the quadrature encoder also input into the module. The module has an integrated connector junction box. In addition, it generates a regulated 5V voltage to power the encoder. The module has channel-to-earth ground double-isolation barrier for safety, noise immunity and channel-to-channel isolation [National Instruments, 2009]. This module is hot-swappable during operation. A partial list of specifications is shown in Table B-5.

[National Instruments, 2009]

Figure B-5: NI-9201 Analog Input Module

Table B-3 : NI-9201 Specifications

[National Instruments, 2009]

Manufacturer	National Instruments	
Model No.	NI-9201	
Parameter	**Value**	**Units**
Analog Inputs	8 (single-ended, 12 bit res)	
Range	±10	V
Sampling Rate	500 (aggregate rate)	kS/S
Temperature	-40 to 70	^{o}C
Connector	Screw terminal / D Sub	

[National Instruments, 2009]

Figure B-6: NI-9411 Digital Input Module

[1] The motor has 6 rotor poles and 8 stator poles and there are four signals (one for each phase A, B, C, D). Each signal occurring six times in each rotation of the rotor

Table B-5: NI-9411 Specifications

[National Instruments, 2009]

Manufacturer	National Instruments	
Model No.	NI-9411	
Parameter	**Value**	**Units**
Digital Inputs	6	
Range	±5 to 24 (differential /single-ended digital input)	V
Sampling Rate	500 (aggregate rate)	ns
Temperature	-40 to 70	^{o}C
Connector	Screw terminal / D Sub	

B.6 Digital Output Module

The NI-9472 digital output module provides extended voltage ranges (up to 60V, 5V TTL) and high current-switching capacity (externally powered, up to 20 A per module) for direct control of a wide array of industrial devices including motors [National Instruments, 2009]. It has an integrated connector junction box with screw-terminals for easy connection of output wires. It is hot-swappable during operation and has channel-to-earth ground double-isolation barrier (isolation for up to 2,300 V_{rms}) for safety and noise immunity [National Instruments, 2009]. It is capable of as low as 100 ns output rate for ultrahigh-speed control, pulse-width modulation (PWM), as well as digital communication. In the test set-up, this module is used to send the digital PWM signals to the driver circuit for each of the four H-bridge amplifiers connected to the motor phases.

[National Instruments, 2009]

Figure B-7: NI 9472 Digital Output Module

Table B-4: NI-9411 Specifications

[National Instruments, 2009]

Manufacturer	National Instruments	
Model No.	NI-9472	
Parameter	**Value**	**Units**
Digital Outputs	8 (100 µs digital o/p)	
Range	±6 to 30 (sourcing digital input, 750 mA)	V
Sampling Rate	100	ns
Temperature	-40 to 70	^{o}C
Connector	Screw terminal / D Sub	

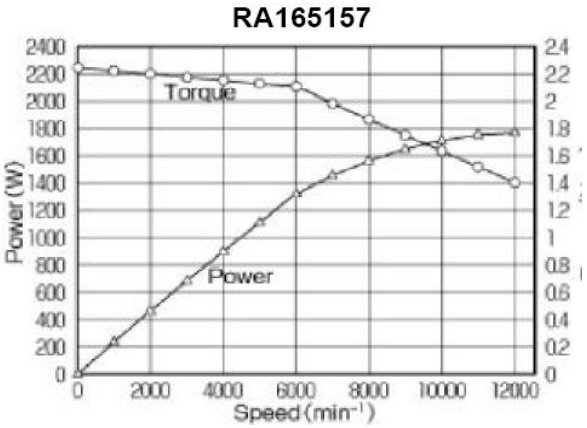
A total of 7 digital signals are needed as output to the four amplifiers. Three of these signals are static and common to all the H-bridges, the remaining four signals control the ON/OFF state of each H-bridge (and hence the associated motor phase). These

low power digital output signals from the module are electrically isolated from the high power side of the motor using a bank of opto-isolators. A single digital PWM signal from the NI-9472 is also converted to an analog DC voltage (using a simple RC low-pass filter in the process) to control the hysteresis brake using the external control capability on the brake power supply/controller. This provides the ability to change the load acting on the motor shaft. Table B-4 provides a partial list of specifications for this module.

B.7 Switched Reluctance Motor

The test motor is a four-phase Switched Reluctance Motor (SRM) manufactured by Motorsoft Inc. A SRM provides various advantages compared to brushed or brushless DC motor. It is more rugged due to the lack of fragile magnets on the rotor or commutation brushes. It is also more suitable for high-speed operation. Since the major source of heating is only the stator, it is easy to provide cooling and enhance the motor performance. The motor rotor consists of tightly stacked laminations of ferromagnetic material. Since all the phases are electrically independent, it is possible to operate the motor with a reduced number of active phases in case of a failure. Thus, the motor is inherently fault-tolerant and provides greater reliability compared to other motor types.

[Motorsoft Inc., 2006]



Power/Torque vs. Speed

Figure B-8: Test Motor Characteristics

The motor has four Hall-effect sensors embedded next to the stator on its non-driving end. These are used to determine the rotor position (as well as motor velocity and acceleration). The rotor position is required for commutation (the sequence in which

motor phases are energized for continuous rotation). The test motor has a relatively low nominal torque (1.8 Nm) but is capable of relatively high-speed operation (6000 rpm).

Table B-6: Switched Reluctance Motor Specifications

Manufacturer	Motorsoft Inc.	
Model No.	RA165157	
Output w/fan and w/out fan	1.13Kw @ 6000 rpm, 0.73Kw @6000 rpm	
Rated Torque	1.8 N-m	
Voltage	24	
Parameter	**Value**	**Units / Notes**
Number of stator poles/rotor poles	8/6	4 phases
Stator pole angle	22.5	deg
Stator outer diameter	130	mm
Stator back iron diameter	104	mm
Stator air gap diameter	70	mm
Air gap length	0.25	mm
Rotor pole angle	23.6	deg
Rotor air gap diameter	69.5	mm
Rotor back iron diameter	52	mm
Rotor bore diameter	25	mm
Total axial length (including winding extension)	70	mm
Total stack length	40	mm
Stacking factor	0.98	
No of laminations	115	
Thickness of lamination	0.35	mm
Lamination Material	S-10	
No of turns per phase	13	
Conductor size	0.8mm,9 parallel	

B.8 Motor Controller

The Open Source Motor Controller (OSMC) is a high-power H-bridge power amplifier designed to control various types of motors in many commercial or industrial control applications [Robot Power Inc., 2009]. It is easy to mount on a variety of platforms and also to interface to a range of microcontroller units. A motor control system can be considered as being split into two distinct levels- the higher level controller which accepts the user input command signals and conveys them to a lower level controller or amplifier that translates these command signals to appropriate motor motion.

The high-power H-bridge amplifier, can be thought of as a dumb controller that receives the PWM signals from the digital output module of the cRIO and transfers them

to the inputs of its bridge driver circuitry. At the heart of the OSMC board is the HIP4081A bridge driver chip. This chip is a monolithic full H-bridge driver chip that includes both high- and low-side FET drivers and all needed voltage boost circuitry [Robot Power Inc., 2009]. It can accept supply voltage ranging from 12-80V and generates all needed signals and voltages to drive the on-board H-bridge. It has four inputs that correspond to the outputs used to switch each individual leg of the H-bridge circuit. There is an additional disable output as well. The input lines of the HIP4081A are modified TTL (a high signal between 3- 12V), which allows a wide variety of signal sources to be used to drive the chip (the present set-up uses 4.5V). The OSMC H-bridge board itself does not have any on-board logic to accept or interpret any user commands. The second part of the system, which provides the external logic to perform this task and generate the PWM signals is a dedicated microcontroller, a PC, etc (or the cRIO in the present case). Although this adds to the overall system complexity somewhat, it provides tremendous flexibility in terms of microcontrollers that can be used to drive the OSMC as well as in the number of motor phases that can be controlled independently [Robot Power Inc., 2009]. The OSMC uses a computer cooling fan instead of a heavy plate type heat-sink to extract heat from the MOSFETs (used for switching). This allows the removal of heat more quickly and allows the controller to be operated at

[Robot Power Inc., 2009]

Figure B-9: Open Source Motor Controller

Table B-7: OSMC Specifications

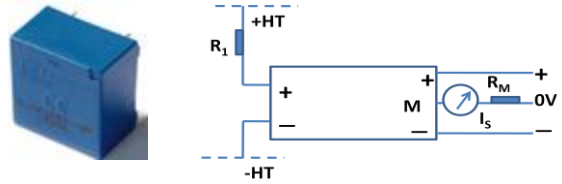
[Robot Power Inc., 2009]

Manufacturer	Robot Power Inc.	
Model No.	OSMC H-bridge v 4.25	
Parameter	**Value**	**Units**
Supply voltage	13-50 (36V max battery rating)	V
Output Current	160 (continuous)	A
MOSFETs	16 ea. IRF1405	
On Resistance	.0026 ohm max at 25C	Ω
Cooling	40 CFM fan	
Logic Interface	10-pin dual-row header	
Power Supply	12V .5A regulator	V, A
Connectors	Solder pads for up to 10 ga wire or #8 bolts	

higher switching frequencies and duty cycles, which in turn directly influences the motor performance characteristics like torque, speed, acoustic noise, etc.

B.9 Voltage Sensors

The LEM Module LV 25-P sensor is a closed loop Hall-effect voltage sensor. It can be used for measuring voltages (DC, AC, pulsed from 10V-500V) with galvanic isolation between the primary circuit (the high current or the motor circuit) and the secondary circuit (the data acquisition set-up). It offers a high accuracy and excellent dynamic performance (response time 40 µs) [LEM Transducers, 2009]. Four voltage sensors are used in the present test set-up and are powered with an independent ±15V DC power supply (Figure B-10). A schematic of the measurement circuit using this sensor is also shown. To measure the voltage applied across the terminals ±HT, a current proportional to the measured voltage, I_{PN}, must be collected through an external resistor R_1, which is installed in series with the primary circuit of the LV 25-P. A large value of R_1 results in a small current flow (I_{PN}) in the primary circuit. This current is transformed by the sensor to produce a proportional current in the secondary circuit, I_S. The proportionality constant between I_{PN} and I_S is called the conversion ratio. The final output voltage is measured across the sensor output (M) and the ground. For the test set-up the values of resistances used are R_1 = 3 kΩ and R_M = 200 Ω (max. input voltage 30VDC). The value for R_M results in a 25mA nominal current for I_S at the maximum output voltage of 5VDC.



[LEM Transducers, 2009]

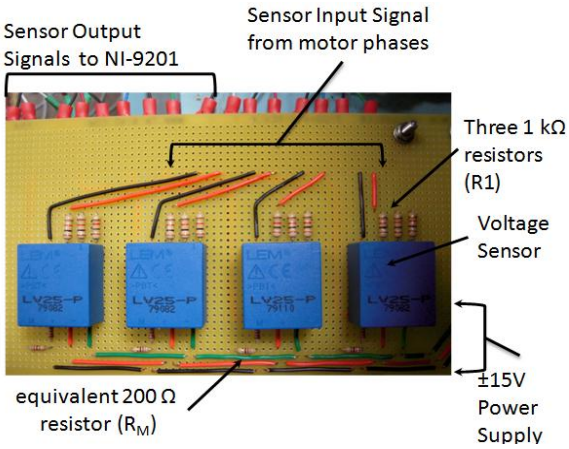


Figure B-10: Voltage Sensor

Table B-8: Voltage Sensor Characteristics

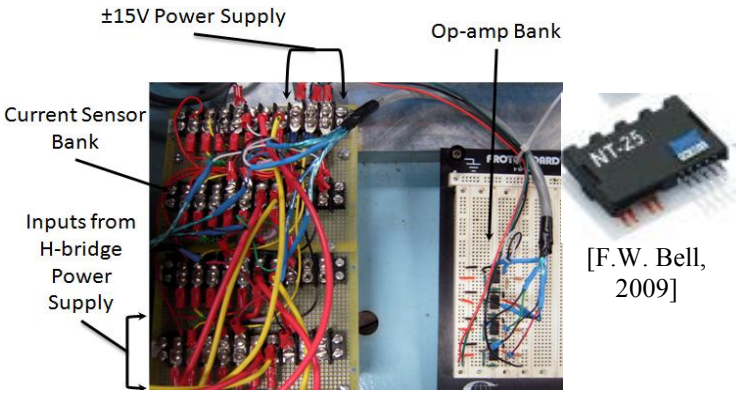
[LEM Transducers, 2009]

Manufacturer	LEM	
Model No.	LV-25P	
Parameter	**Symbol**	**Units**
Primary nominal current	I_{PN}	10 mA
Primary current measuring range	I_P	0 ±14 mA
Sec. nominal r.m.s. current	I_{SN}	25 mA
Conversion ratio (I_S/I_P)	K_N	2500:1000
Overall accuracy	X_G	±0.9%
Linearity	ε_L	<0.2%
Supply Voltage	V_S	±15V

The maximum voltages were selected to suit the NI-9201 module and the maximum input voltage based on the H-bridge rating[2]. The sensor is highly linear when in range. The sensor output is further scaled by a factor of 6 (30VDC/5VDC) with software signal conditioning.

B.10 Current Sensors

The NT Series current sensor can accurately measure both DC and AC currents. It provides electrical isolation between the circuit where the current is measured and the sensor output connected to data acquisition components. These sensors are based on the anisotropic magneto-resistive (AMR) effect (change in circuit resistance with a change in magnetic field). The sensing elements are thin film permalloy magneto-resistors on a silicon substrate. This provides a compact design for mounting directly on circuit boards. A Wheatstone bridge of four MR resistors is used internally in the sensor to ensure that external magnetic fields or sensitivity to temperature do not distort measurement by changing the resistance. An electrically isolated aluminum compensation conductor is integrated on the same substrate above the permalloy resistors [F.W. Bell, 2009]. The current flowing in it generates a magnetic field, compensating that



[F.W. Bell, 2009]

Figure B-11: Current Sensor

[2] These values are chosen only for testing purposes. They can be changed based on requirements.

of the conductor to be measured. The current in the compensation conductor is proportional to measured field amplitude; the voltage drop across a resistor forms the output signal, which is amplified by on-board signal processing.

Four sensors are used in the present test set-up, using the same ±15V DC power supply used for the voltage sensors (Figure B-11). The test bed currents have been restricted to an upper limit of 10A. Since the sensor exhibits highly linear behavior in this range, the voltage output at 10A should ideally be 1V after accounting for the zero current offset. To calibrate the output voltage to represent the current accurately, it is scaled by a factor (I_{PN}/V_{PN}) minus the zero current offset. To fully utilize the scale and resolution of the NI-9201 module (and to account for measurement noise), additional signal processing is performed by a LM741CN op-amp that amplifies the signal up to the level of its supply voltage. The amplification is a function of the external resistors in the op-amp circuit, R_1 = 2.2 kΩ and R_2 = 15 kΩ. The amplification ratio is given by $1+(R_2/R_1)$. This effectively extends the expected output voltage to around the range of 0-8V. This range is lower than the 15V op-amp power supply and hence voltage saturation is not a major concern.

Table B-9: NT-25 Current Sensor Characteristics

[F.W. Bell, 2009]

Manufacturer	F.W.Bell	
Model No.	NT-25	
Measuring Circuit	**Units**	**Value**
Nominal current, I_{PN}	±A	25
Peak current	±A	75
Output at nominal, V_{PN}	±V	2.5
Supply voltage	±V	12 to 15
Linearity (at +25¼C)	±%	0.1
Typical zero offset (at +25¼C)	±mV	7.5
Max. offset over temperature	±mV	35`
Operating temperature range	°C	-25 to 85
Weight	grams	4.5

B.11 Torque Sensor

Indirect methods of deriving motor torque (using analytical relations) from other prime mover electromagnetic characteristics do not account for perturbations due to inherent ripple in the torque profile caused by switching of the motor phases and external load or disturbance torques. A direct measurement of the motor torque is therefore preferable, using a rotary torque sensor. In most rotary torque sensors, the shaft torque is typically measured by either sensing actual shaft deflection due to torque, or by detecting

the effects of deflection using sensing elements directly bonded to the shaft. Torque sensors are therefore serial in nature in that they have to be placed directly in the torque path. The torque sensor in use on the present test set-up is a Lebow1701-2 inline rotary torque transducer, mounted between the motor and load (hysteresis brake). The sensor uses strain gages in a Wheatstone bridge arrangement to measure the motor torque. The gages are inductively coupled to stationary windings on the assembly by a rotary transformer, for power and signal return. As the shaft deforms due to applied torque, the resistance in the bonded foil strain gages changes and modulates the applied input voltage. The change in the resistance of the bridge is converted to a calibrated output signal directly proportional to the applied torque.

[Honeywell Transducers, 2009]

Figure B-12: Torque Sensor

Table B-10: Torque Sensor Specifications

Manufacturer	Honeywell Transducers	
Model No.	1701-2	
Parameter	**Value**	**Units**
Accuracy	0.25	%
Built-in instrumentation amplifier	±10	V
Range	2 (Overload=200% rated output)	Nm
Operating temperature	0-60	^{0}C
Supply Voltage	12 (±10 %)	V
Limit frequency (3dB)	200	Hz
Voltage output	0 to ±10	V
Spring constant	248	Nm/rad
Maximum permissible loads	77 (radial and axial)	N
Mass moment of inertia	2,11	g/cm^2

The bridge configuration helps to compensate for temperature effects and extraneous signals arising from secondary loads on the shaft. The contactless transmission of supply voltage and measuring signal enables continuous operation with low maintenance. The sensor specifications in Table B-10 were verified by performing an experimental calibration after integrating the sensor into the test-bed. The torque sensor used in the present case is rated for a load of 2 Nm and an output voltage range of ±10V. The sensor is powered using a dedicated 12V, 500 mA power supply. The torque/voltage

curve is linear for the entire range of operation. The output from the torque sensor is connected to the NI-9201 analog input module.

B.12 Position Sensor

The Hall-effect sensors serve as commutation sensors for high-speed operation. In such cases, using a high-resolution position sensor is sometimes impractical due to bandwidth requirements. On the other hand, for implementing advanced commutation strategies like modifying the phase turn-on/off angles for shaping the torque profile, reducing the torque ripple, reducing acoustic noise, etc. there is a need to infer the rotor position with a high resolution (of the order of tenths of degrees). To provide this capability on the test-bed, a high resolution optical encoder has been used. The E6 series rotary encoder uses a transmissive optical encoder module (stationary) with a removable 10-pin finger-latching connector. The module consists of a highly collimated solid-state light source and monolithic phased array sensor (detector), which together provide a system extremely tolerant to mechanical misalignments [U.S. Digital Inc., 2009]. The rotary component consists of a shatterproof, transparent mylar disk mounted to a precision-machined aluminum hub on the motor output shaft and aligned with the encoder module. The number of line pairs patterned on the outer edge of the disc dictates the resolution. Power is supplied from a single +5VDC dedicated power supply. This quadrature encoder has two channels, *A* and *B*, electrically offset from one another by ¼th of a cycle (90^0) to infer direction (*B* leads *A*=>clockwise, *A* leads *B*=>counterclockwise). Both these outputs are connected to the NI-9411 module. A benefit of this scheme is the ability to electronically multiply encoder count. In the x1 mode all counts are generated on rising edges of *A*. In the x2 mode, both rising and falling edges of *A* generate counts. In the x4 mode, the rising and falling edges of both *A* and *B* generate counts, increasing resolution fourfold.

[U. S. Digital Inc., 2009]

Figure B-13: E6 Incremental Encoder

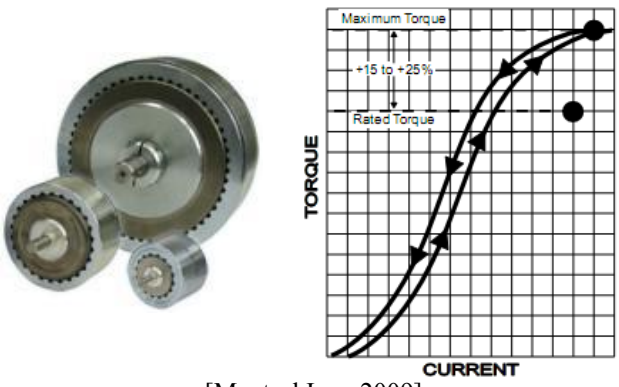
The index channel outputs one pulse-per-rotation equal to ¼th of a quadrature cycle. Output complement signals can also be used as differential inputs for improved common mode noise rejection. The digital signals are also used to calculate motor velocity and acceleration values through successive discrete differentiation in the FPGA software.

Table B-11: Incremental Encoder Specifications

Manufacturer	U.S. Digital Inc.	
Model Number	EM-6 kit Incremental Encoder with HEDS read-head	
Parameter	**Value**	**Units**
Resolution	~ 9 (4x after quadrature)	Bits
Supply Voltage	4.5-5.5	V
Count Frequency	100 kHz (rpm/60)x cycles/rev	kHz
Operating Temperature	-40 to 100	°C
Output Voltage	2.4-3.4 (sourcing to +5V)	V

B.13 Hysteresis Brake

Hysteresis brakes produce a braking torque through a magnetic air gap without any magnetic particles or friction components. This provides better operating characteristic like smoother torque profile, longer life, better reliability, controllability, etc. Magnetic hysteresis effect is used for torque control via a reticulated stator pole structure and a steel rotor/shaft assembly. Until the field coil is energized, the drag cup spins freely on its bearings. When a magnetic field is applied to the pole structure, the rotor is magnetically restrained due to the air gap flux field, creating a braking action [Magtrol Inc., 2009]. Since braking torque is produced purely by magnetic fields, the brake load can be finely controlled independent of speed by controlling the DC current supplied to the field coil. The polarity of current does not affect the braking effect. Due to non-contacting active components, brake operation is



[Magtrol Inc., 2009]

Figure B-14: Magtrol Hysteresis Brake

practically silent and smooth at any speed (very high speed operation is possible) [Magtrol Inc., 2009].

There is no wear and tear of components, particle emission, seal leakage, etc. all of which combine to provide an exceptional brake life. These factors also provide better torque repeatability, which is needed for testing purposes. A typical hysteresis brake and its torque vs. current characteristics are shown in Figure B-14. The HB 140 M-2 brake used on the testbed is controlled with a combined Magtrol-5210 power supply and controller. The controller has a 10-turn potentiometer with 3 ranges for high-resolution manual current control. It also offers external control via a 0-5V signal using a 3.5mm TRS plug (for instance, using a PWM output from the NI-9472 module). This functionality permits precisely controlled load profiles to be applied automatically during testing along with static loads when needed.

Table B-12: Hysteresis Brake Specifications

[Magtrol Inc., 2009]

Manufacturer	Magtrol Inc.	
Model No.	HB 140 M-2	
Parameter	**Value**	**Units**
Min. Torque at Rated Current	1	Nm
Rated Current	253	mA
Voltage	24	V
Max. Speed	12000	rpm
Kinetic Power	300 (5min)/75 (cont.)	W

B.14 Acoustic Noise Sensor

Acoustic energy in the range 20 Hz - 20 kHz is manifested as audible sound that can be perceived by human hearing. Sound is perceived and measured as pressure fluctuations relative to the ambient pressure. Acoustic noise transducers like microphones measure frequencies in the audible range by transforming such pressure oscillations into electrical signals. The Sound Pressure Level (SPL) is calculated as:

$$SPL = 20 \log (P / P_{ref}) \text{ dB} \qquad \text{Eq. B-1}$$

where P, P_{ref} are the measured and reference pressures respectively (P_{ref} =20μPa). The sensitivity is often specified in units of mV/ Pa of sound pressure. The range and frequency response are crucial when selecting an appropriate sensor. The range of SPLs for which a sensor meets its performance specifications is limited on the low end by the inherent system noise and on the high end by the maximum SPL (where total harmonic

distortion reaches 3%) dictated by its design and physical characteristics. The frequency response is the range of frequencies for which the sensor maintains a constant sensitivity within defined limits. Ideally, it should sufficiently cover the frequency bandwidth range required for measurement and be as flat as possible in this range.

Table B-13 : Digital Sound Level Meter Specifications

Manufacturer	Radioshack	
Model No.	33-2055	
Parameter	**Value**	**Units**
Microphone	Electret Condenser	
Range	50 to 126	dB
Accuracy	± 2 at 114 dB SPL	dB
Reference	0 dB	0dB = 0.0002 MicroBar
Weighting	A and C	
Output	1 (Peak to Peak, Open Circuit, F.S. @ 1 kHz)	V
Impedance	10 (Min. Load)	kΩ
Distortion	Less than 2% at 1 kHz, 0.5 V p-p Output	

The sensor used for the present test set-up is a RadioShack digital sound level meter, Model 33-2055. The meter consists of a front-mounted microphone and switches for changing the weighting and speed of response to sound pressure changes. The weighting filters (*A*, *B*, *C*, *D*, *Z*) are a standard family of curves defined in the International standard IEC 61672:2003. They reduce the contribution of low and high frequencies to produce a reading that corresponds approximately to what is perceived by human hearing. The weighting switch allows for switching between standard *A* and *C* weightings. *C* weighting provides more or less uniform sensor response between 32 Hz -10kHz, and *A* weighting makes it more sensitive to frequencies in the range 0.5-10kHz (also the most sensitive range of the human ear). The response switch allows for changing the speed of the meter's response. The slow setting makes it less sensitive to rapid changes in sound level and can be used

[Radioshack, 2009]

Figure B-15 : Digital Sound Level Meter

for measuring average pressure levels. Conversely, the fast setting is more useful for rapid changes in SPLs and when peak sound levels are measured.

B.15 Vibration Sensor

A typical method of measuring motor vibration is by using an accelerometer. Piezoelectric accelerometers are commonly used in many vibration monitoring applications since they can sense much higher frequencies as compared to competing technologies like strain gauge based sensors. A piezoelectric material (like quartz or polycrystalline ceramics like PZT) produces an electric potential when it experiences mechanical stress in the form of compressive or tensile forces, shear forces or flexural forces. In a piezoelectric accelerometer, a seismic mass is placed in the accelerometer housing in such a manner that stresses the piezoelectric sensing element when the mass is accelerated. The generated electric potential is the basis of the accelerometer output signal and is directly proportional to the vibration forces. These sensors have a broad frequency response ranging from a few Hz to many kHz. They are self-generating and do not need a separate power source. However, most sensors include some built-in signal processing electronics which require a power supply. Signal conditioning is required to convert the high impedance charge output from the piezoelectric sensing element to a low impedance voltage output that can be transmitted over cables with relatively good immunity to signal loss or noise and for easy interfacing with any data acquisition system.

[PCB, 2009]

Figure B-16: PCB Piezotronics 353B17 Accelerometer

The sensor used on the test bed is a PCB Piezotronics ICP Accelerometer Model 353B17 (the term ICP indicates that the accelerometer has a built-in signal amplifier). This sensor has 10mV/g sensitivity over a 0-500g range. The single axis sensor measures vibrations only along the radial axis of the motor. The accelerometer is mounted with a threaded stud on the motor casing as it provides excellent coupling between the sensor

surface and the test object. The test bed uses a 482A22 ICP signal conditioner from PCB Piezotronics that provides the required constant current (adjustable, 0-20mA) for the signal amplifier. The unit can power up to four accelerometers simultaneously.

Table B-14: Accelerometer Specifications

[PCB, 2009]

Manufacturer	PCB Piezotronics	
Model Number	353B17	
Parameter	**Value**	**Units**
Sensitivity (± 10%)	10	mV/g
Measurement Range	± 500	g
Freq. Range (± 5%)	1-10,000	Hz
Resonant Frequency	≥ 70,000	Hz
Non-Linearity	≤ 1	%
Overload Limit (Shock)	± 10,000 pk	g
Excitation Voltage	18 to 30	VDC
Const. Current Excitation	2 to 20	mA
Sensing Element	Quartz	

Additionally, the signal-conditioning unit minimizes the need for specialized low noise cables and converts the accelerometer to a signal that can be read by the NI-9201 module. Due to the low sensitivity of the sensor and the relatively low accelerations experienced by the motor, additional signal amplification has been implemented. Using a LM741CN op-amp with a 47 kΩ resistor and a 510Ω resistor, a signal gain of approximately 93 is produced. Additionally, a 1 kΩ resistor is used as a pull down resistor to eliminate signal drift. The sensor output is further analyzed in software using Fast Fourier Transforms (FFT) to determine the vibration frequencies.

Appendix C. The Actuator Management and Operation Software (AMOS) Framework

Most commercially available EMAs are driven using a motion controller with proprietary embedded software for motor commutation, sensor data processing, communication and control. However, such software is typically custom–designed for a specific type of EMA used in an application and not portable across a spectrum of EMAs (with different types of prime movers, geartrains, sensors, etc.) as described in the Electromechanical Actuator Architecture (EMAA) [Tesar, 2009] or across different application domains in which these EMAs may be used. The execution platforms for the software are also often tied to specific hardware or operating systems. An effort has therefore been initiated at the UT-RRG to formalize a more universal software framework for intelligent EMAs, termed as the '*A*ctuator *M*anagement and *O*peration *S*oftware (*AMOS*)' along the lines of *OSCAR* (*O*perational *S*oftware *C*omponents for *A*dvanced *R*obotics) [Kapoor and Tesar, 1996], a system-level software developed for robotic manipulators. The initial architecture for *AMOS* was proposed by [Huang and Tesar, 2000] and later refined by [Yun and Tesar, 2007]. Considering code extensibility and reusability, the object-oriented style of software development provides many advantages by via abstraction, encapsulation, inheritance, etc. and has been deemed suitable for the detailed design of sub-components of *AMOS* [Yun and Tesar, 2007].

The goal is to develop a framework of C++ libraries that can be used for creating supervisory control programs for a spectrum of intelligent EMA applications. This provides the flexibility to create individual modules of the framework independently during the software development, debugging and testing phase as well as provides the end-user of the final product the ability to choose requisite modules to develop customized programs for addressing application-specific needs. This includes incorporating features like low-level commutation and control for different types of prime movers, data acquisition and signal processing for different analog/digital sensors, control of auxiliary devices like clutches, brakes, etc.; algorithms like Condition-Based Maintenance (CBM) algorithms [Demling and Tesar, 2006][Hvass and Tesar, 2004], the

Sensor and Process Fault Detection and Isolation algorithm developed in this report; functionality for display, combination and manipulation of decision-surfaces [Yun and Tesar, 2007] [1], system-level operational decision making [Ashok and Tesar, 2007], etc. The components of *AMOS* can be broadly categorized into three levels based on loop update frequency requirements [Yun and Tesar, 2007]: the management level, the control level and the sensor and communication level, with the loop rates increasing from top to bottom (Figure C-1)

At the *Management* level AMOS receives input commands from the end-user/system operator via a graphical user interface (GUI) or from system-level software like *OSCAR*. Typical inputs may be a desired position, speed, torque, or noise level. These commands are then processed; along with a combination of the stored decision surfaces, measured sensor data, and user-specified performance criteria, to yield the appropriate control signals for operation. This level therefore contains code for combining the decision surfaces, calculating the norms for those surfaces, displaying the different maps/envelopes and norms on the GUI, CBM, etc.

The code contained at the *Control* level is primarily responsible for all the interactions with the hardware where the user inputs are translated into "real" commands (by modulating the control parameters like current or voltage) to control the EMA. This level is also responsible for the control of ancillary devices like brakes, lubricant/cooling system etc. The lowest *Sensors and Communication* level includes code for communication among the AMOS modules, between the EMA and the system level controller, etc. This module is also responsible for the data acquisition and processing of data from all the sensors in the EMA and ensuring that the highest quality data is available at all times for different levels of the control hierarchy. An efficient commutation routine for a four phase switched reluctance motor has been developed as

[1] [Yun and Tesar, 2007] described the creation of a graphical user interface for *AMOS* with features such as pan, rotate, zoom, etc. for better visual display, clicking on specific operational points on a performance map to automatically command the EMA control parameters, setting weights for the different performance criteria, combining performance criteria, etc.

part of the control module during the course of this work (Figure C-2). The SPFDI algorithm developed in this work has been implemented as part of the sensor module.

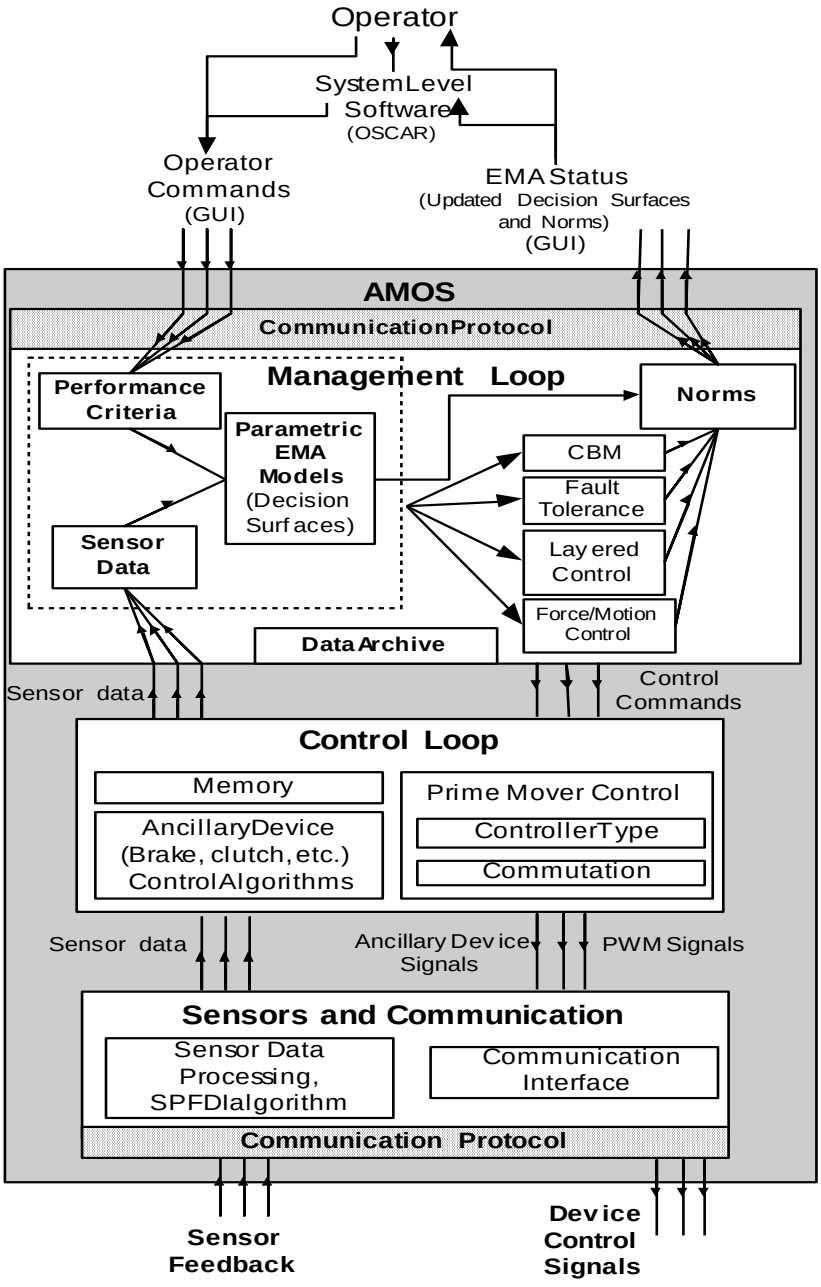


Figure C-1: The AMOS Framework

The code developed for this work uses a combination of C++ and LabView™ virtual instruments (VI)[2]. As shown in Figure C-4, the VIs are used to display control parameters that can be changed by the user as well as record and display measurement data obtained from all the sensors integrated into the testbed (Figure C-3). The independent control parameters presently available are the main line voltage (controlled directly from the variable voltage power supply), the Pulse Width Modulation (PWM) switching duty cycle and frequency for each motor phase and the phase turn-on/off angles for each phase. The sensor data obtained is filtered using a combination of hardware and software filters to remove extraneous noise and is used to generate performance maps and derive the CPT parameters for the Bayesian network used for the SPFDI algorithm. C++ has been used to develop code to implement a basic graphical user interface that can be used to create Bayesian networks (Figure C-5), as well as perform exact inferencing (using Pearl's belief propagation algorithm, Chapter 2, Section 2.5.2.1.1) or approximate inferencing (using Logic Sampling and Likelihood Weighting algorithms as described in Chapter 2, Sections 2.5.2.2.1. and 2.5.2.2.2. respectively). In addition, the logic for the SPFDI algorithm (Chapter 5, section 5.4) has also been implemented in C++. The exchange of data between the C++ and LabView sections of the code is carried out using simple text files. All the VIs are written for use with LabView 8.6 using the cRIO-9112 chassis with FPGA connected to a NI-9014 Real Time Controller (RTC). A host computer (PC) is used to interface with the cRIO.

With the present hardware configuration, there are 3 processors available to run the VIs on: the FPGA, the RTC, and the PC. Hence different portions of the code can be executed on different platforms In general, processes that need to be deterministic and fast such as data acquisition and control are programmed on the FPGA. Additionally, all communication through the I/O modules (for the motor, brake and sensors) is handled by the FPGA. Processes that are relatively less time critical such as signal conditioning, data processing, and data logging are programmed to run on the RTC. Programs running on the PC are primarily used to transfer data from the built-in hard drive on the cRIO to the

[2] This was done to ease the development of preliminary code. The eventual goal is to have all the program components in C++ for greater portability and flexibility

PC hard drive for use with other components of *AMOS* like the graphical user interface, Bayesian network inferencing, etc. The SPFDI code is also executed on the PC.

The testbed software is written such that the relevant testing parameters (such as the PWM values, brake load, etc.) are communicated from the *AMOS* interface running on the PC to a top level VI running on the RTC that in turn communicates with a top level VI running on the FPGA. All the top level RTC VIs primarily serve the same purpose i.e. accept user specified test parameters and control the actual tests. The test parameters may be input manually or via text files. The FPGA VI controls the motor and the brake via signals from the cRIO output modules. This VI has the functionality to use either the Hall Effect sensors embedded in the motor or the digital encoder to dictate the motor commutation. For all the tests carried out in the present work, the motor phases are turned on at an angle of -30° and turned off at an angle of -15° (with respect to the completely aligned position of the rotor and stator poles). When using the Hall-effect sensors for commutation, these angles are fixed. Using the digital encoder allows the user to specify the turn on and turn off angle for the phases for smoother torque profiles and better acoustic noise control.

The FPGA VI also acquires the sensor data by reading signals from the cRIO input modules. The data is organized and queued for reading by the RTC. The top level RTC VI then reads the sensor data from the FPGA queue, and calibrates it for display and data logging. Numerous other sub-VIs are used to in addition to these top level VIs and are not discussed in detail here. A typical front panel for a RTC VI is shown in Figure C-4. VIs running on the PC serve to transfer data from the RTC to the PC via ethernet using the FTP capability built into RTC.

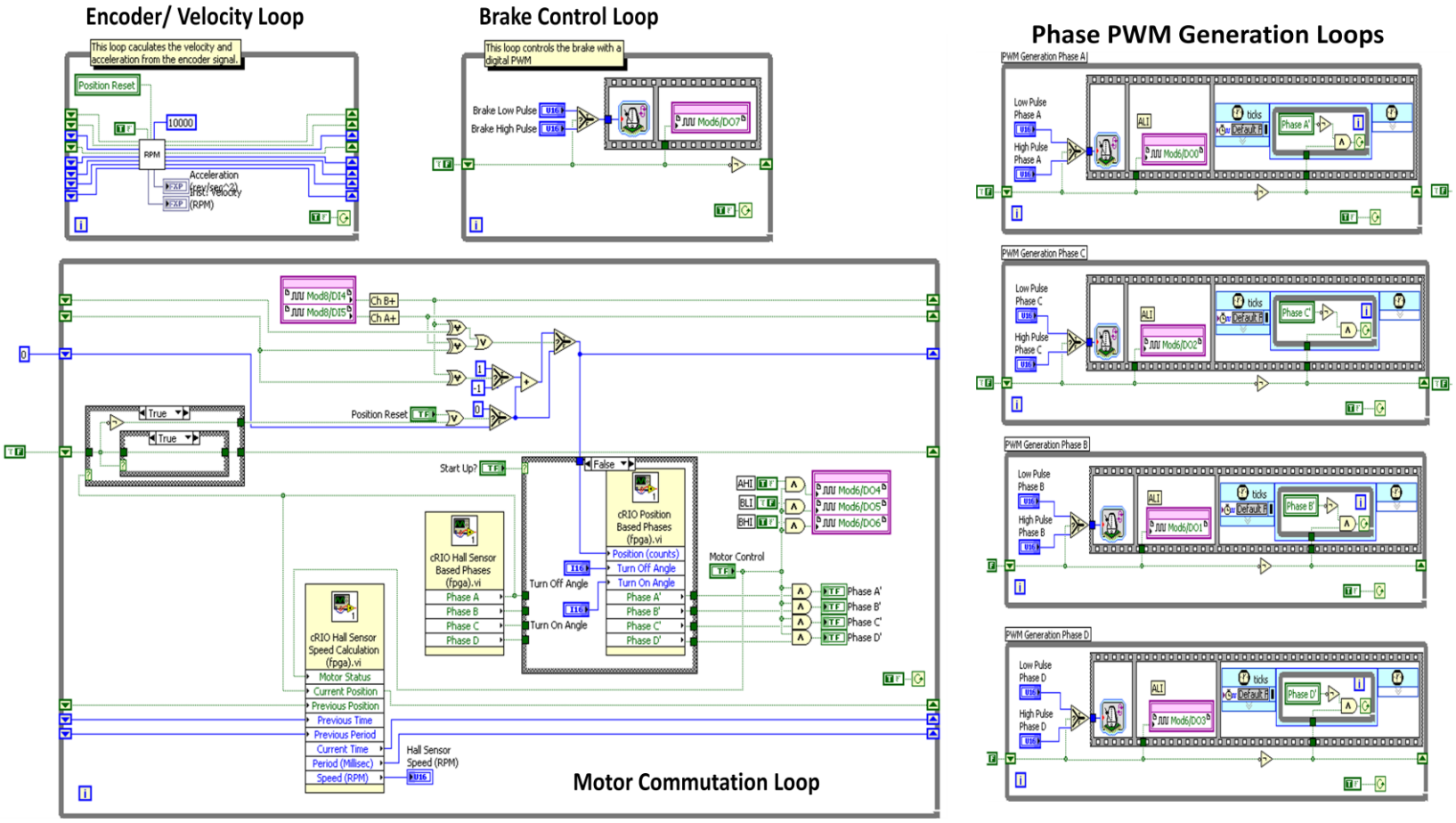


Figure C-2: Commutation Routine for a Four Phase Switched Reluctance Motor

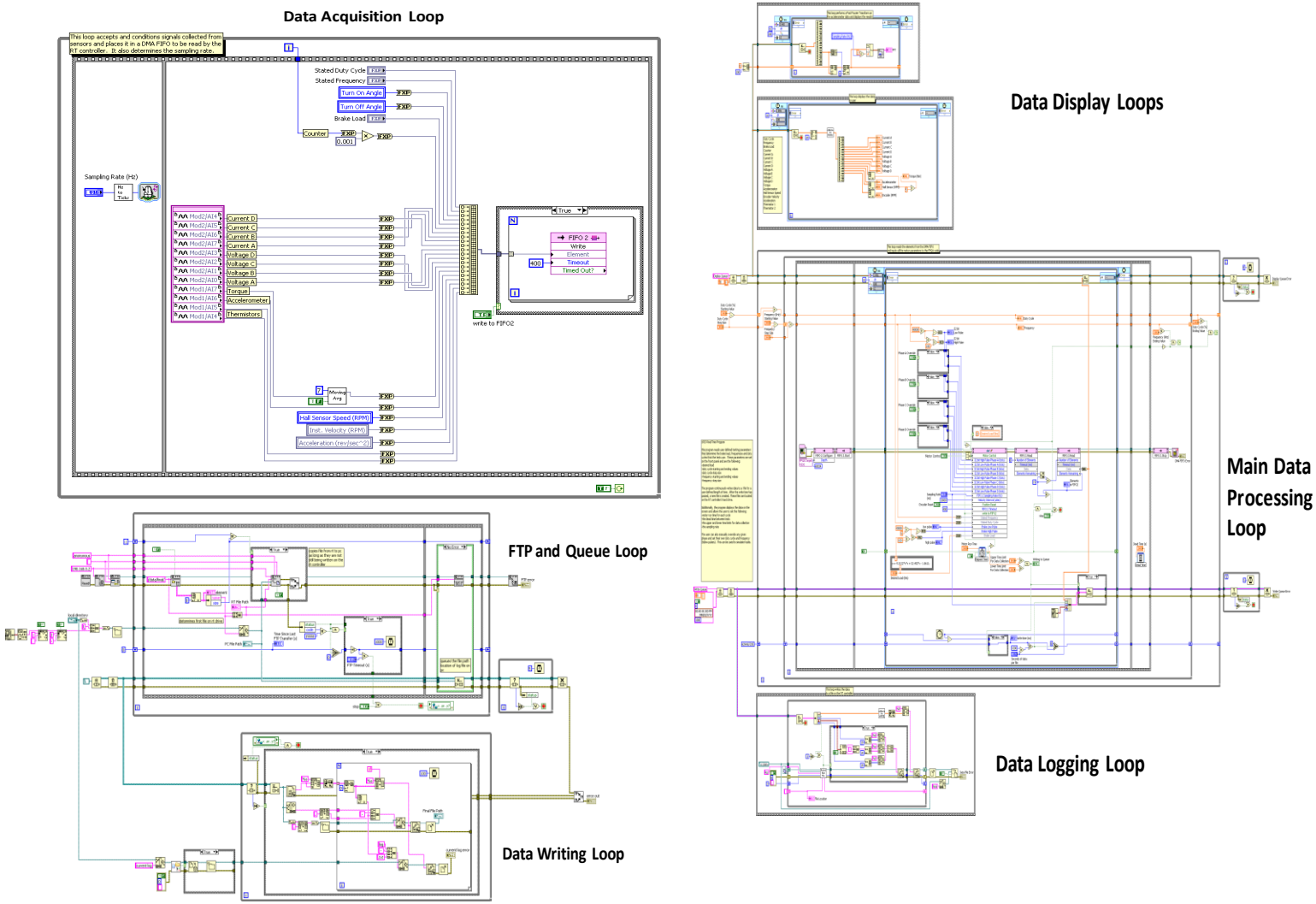


Figure C-3: LabView code for Sensor Data Acquisition and Data Logging

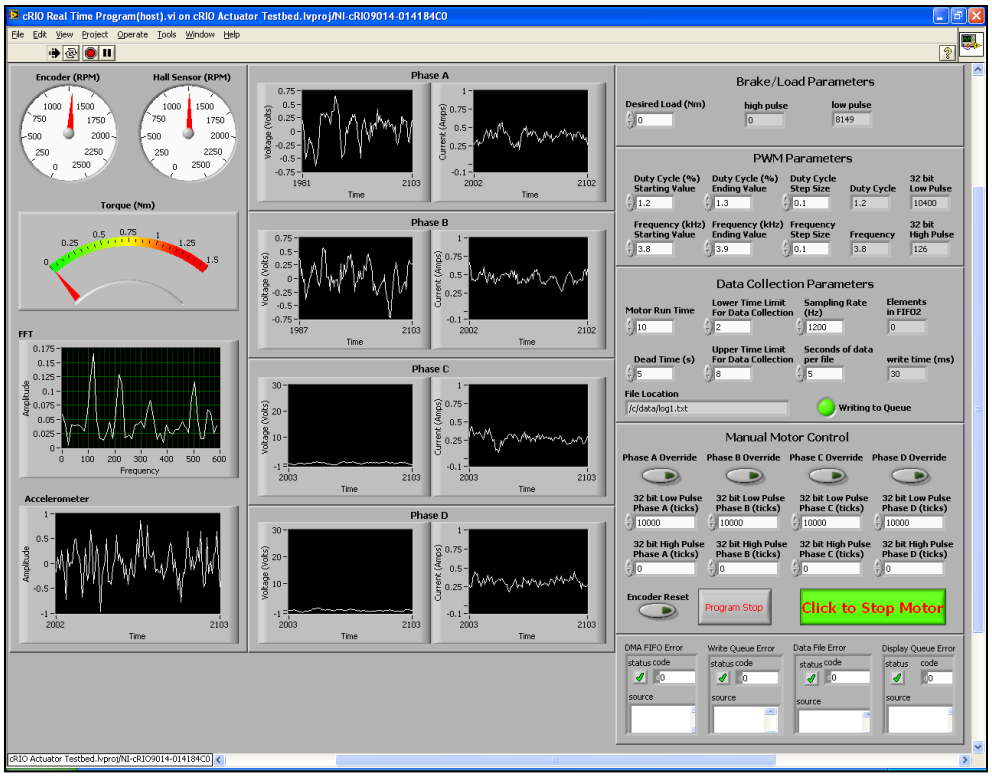
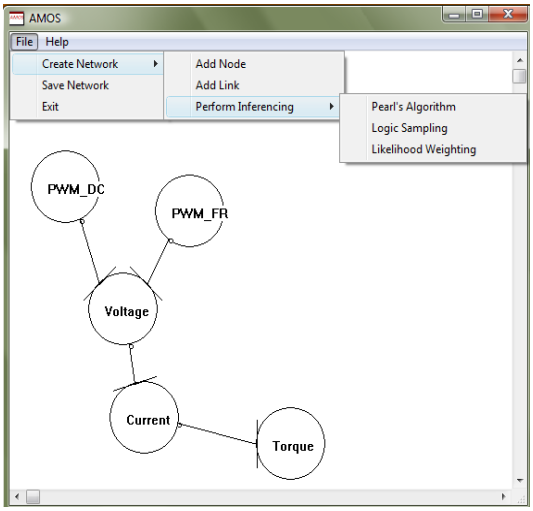


Figure C-4: Front Panel of a top level LabView Real Time Controller VI



Preliminary GUI for Bayesian Networks

AMOS Command Line Interface

Figure C-5: AMOS User Interface

References

Abdelghani, M., and Friswell, M.I, 2007, "*Sensor Validation for Structural Systems with Multiplicative Sensor Faults*," Mechanical Systems and Signal Processing, pp. 270-279, Vol. 21.

Acid, S., de Campos, L.M., Fernandez-Luna, J.M., Rodriguez, S., Rodriguez, J. M., and Salcedo, J. L., 2004,"*A Comparison of Learning Algorithms for Bayesian Networks: A Case Study Based on Data from an Emergency Medical Service*," Artificial Intelligence in Medicine, pp. 215–232, Vol. 30.

Alag, S., Aggogino, A.M., and Morjaria, M., 2001, "*A Methodology for Intelligent Sensor Measurement, Validation, Fusion, and Fault Detection for Equipment Monitoring and Diagnostics*," Artificial Intelligence for Engineering Design, Analysis and Manufacturing, pp. 307–320, Vol. 15.

Allen, D. and Darwiche, A., 2003, "*Optimal Time-Space Tradeoff in Probabilistic Inference*," Proc. of the 18th Intl. Joint Conf. on Artificial Intelligence (IJCAI-03), pp. 969-975, Acapulco, Mexico, August 9-15.

Alridge, H.A., and Juang, J.N., 1997, "*Experimental Robot Position Sensor Fault Tolerance Using Accelerometers and Joint Torque Sensors*," NASA Technical Memorandum 110335, pp. 14, March.

Anwar, M. N., and Husain, I., S, 2000, "*Radial Force Calculation and Acoustic Noise Prediction in Switched Reluctance Machines*," IEEE Trans. on Industry Applications, pp. 1589-1597, Vol. 36, No. 6, November / December.

Aradhye, H., and Heger, S.A., 2002, "*Signal Validation Using Bayesian Belief Networks and Fuzzy Logic*," in Fuzzy Logic and Probability Applications: Bridging the Gap, Eds. Ross, Timothy, J., Booker, J.M., Parkinson, J.W., ASA-SIAM Series on Statistics and Applied Probability.

Aradhye, H., 1997, "*Sensor Fault Detection, Isolation, and Accommodation Using Neural Networks, Fuzzy Logic, and Bayesian Belief Networks*," Master's thesis, University of New Mexico.

Ashok P., and Tesar, D., 2007, "*Math Framework for Decision Making in Intelligent Electromechanical Actuators*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Ashok, P., and Tesar, D., 2001, "*Design Synthesis Framework for Switched Reluctance Motors*," Master's Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Aughenbaugh, J.M, and LaCour, B.R., 2008, "*Metric Selection for Information Theoretic*

Sensor Management," Proc. of the 11th Intl. Conf. on Information Fusion, pp.1-8.

Balamurugan, S., and Sumathi, P., 2004, "*Analysis of Temperature Rise in Switched Reluctance Motor Due to the Core and Copper Loss by Coupled Field Finite Element Analysis*," Proc. of the IEEE Intl. Conf. on Power System Technology (POWERCON 2004), pp. 630-634, November.

Bauer, E., Koller, D., and Singer, Y., 1997, "*Update Rules for Parameter Estimation in Bayesian Networks*," Proc. of the Thirteenth Annual Conf. on Uncertainty in Artificial Intelligence, pp. 3-13.

Bedaux, J., 2008, C++ port code for Mersenne Twister Pseudo-random Number Generator, available online at http://bedaux.net/mtrand/

Berchialla, P., Cecilia Scarinzi, C., Snidero, S., and Gregori, D., 2007, "*Comparing Adaptive Bayesian Network, Artifickal Neural Networks, Classification Trees and Classical Logistic Models in Quantitative Risk Assessment: the Case Study of Foreign Body Injuries in Children*," I2C Research Group Working Papers, 04/2007.

Bickmore, T.W., 1994, "*Real-Time Sensor Data Validation*," Gencorp, Aerojet Propulsion Division, NASA Contractor Report No. 195295.

Benhama, A., Williamson, A.C., and Reece, A.B.J., 1999, "*Force and Torque Computation from 2-D and 3-D Finite Element Field Solutions,*" IEEE Proc. on Electric Power Applications, pp. 25-31, Vol. 146, No. 1, January.

Beno, M.M, Marimuthu, N.S., and Singh, N.A, 2007, "*Nonlinear Modeling of Switched Reluctance Motors Using Soft Computing Techniques*," Intl. Journal of Electrical and Power Engineering, pp. 215-221, Vol.1, No. 2.

Bladon, P., Day, P.S., Hughes, T., and Stanley, P, 2006, "*High-Level Fusion using Bayesian Networks: Applications in Command and Control*," Information Fusion for Command Support, Proc. RTO-MP-IST-055, Paper 4, pp. 4-1–4-18, France.

Bluethmann, B., Herrera, E., Hulse, A., Figuered, J., Junkin, L., Markee, M., and Ambrose, R., 2010, "*An Active Suspension System for Lunar Crew Mobility*," IEEE Aerospace Conf., Big Sky, Montana, March 6 -13.

Boerlag, B., 1993, "*Link Strength in Bayesian Networks*," Master's Thesis, Department of Computer Science, The University of British Columbia.

Borghetti, B.J., 1996, "*Inference Algorithm Performance and Selection Under Constrained Resources*," MS Thesis, Air Force Institute of Technology Air University (Document no. AFIT/GCS/ENG/96D-05), December.

Borsotto, M., Savell, T.C., Reifman, J., Hoyt, R.W., Nunns, G. and Crick, C.J., 2004, "*Life-Signs Determination Model for Warfighter Physiological Status Monitoring*," NATO Technical Report RTO-MP-HFM-109 , Proc. of RTO

Human Factors and Medicine Panel (HFM) Symposium, Paper 28, Florida, August 16-18th.

Brancato E.L, 1992, "*Estimation of Lifetime Expectancies of Motors*," IEEE Electrical Insulation Magazine, pp. 5 – 13, Vol. 8, No. 3, May/June 1992.

Brignell, J., and White, N., 1996, "*Intelligent Sensor Systems*," Inst. of Physics Publ., UK

Brooks, R. R., and Iyengar, S. S., 1998, "*Multi-Sensor Fusion: Fundamentals & Applications*," Prentice Hall.

Brown, D., Georgoulas, G., Bae, H., Vachtsevanos, G., Chen, R., Ho, Y. H., Tannenbaum, G., and Schroeder, J.B., 2009, "*Particle Filter Based Anomaly Detection for Aircraft Actuator Systems*," Proc. IEEE Aerospace Conf., pp.1-13.

Bruemmer, D., Few, D., Boring, R., Marble, J., Walton, C. and Nielsen, C., 2005, "*Shared Understanding for Collaborative Control*," IEEE Trans. on Systems, Man and Cybernetics –Part A: Systems and Humans, pp. 494–504, Vol. 35, No. 4.

Burns, P.G., 2009, "*The Combat-Wireless Health Monitoring System*," CrossTalk, The Journal of Defense Software Engineering, pp. 4-9, December.

Byington, C.S., Watson, M., and Edwards, D., 2004, "*Data-Driven Neural Network Methodology to Remaining. Life Predictions for Aircraft Actuator Components*," Proc. of the IEEE Aerospace Conf., pp. 3581 – 3589, Vol. 6, March 6-13.

Cai, W., Pillay, P., Tang, Z., and Omekanda, A., 2001, "*Experimental Study of Vibrations in the Switched Reluctance Motor*," IEEE Intl. Conf. on Electric Machines and Drives IEMDC 2001, pp. 576-581.

Camarillo, D.B., Krummel, T.M., and Salisbury, Jr., J.K., 2004, "*Robotic Technology in Surgery: Past, Present, and Future*," The American Journal of Surgery, pp. 2S–15S, Vol. 188, October.

Castanon, L.E.G., Gonzalez, J.P.N, Castanon, M.A.G., and Morales-Menendez, R., "*Fault Diagnosis of Industrial Systems with Bayesian Networks and Neural Networks*," MICAI 2008 Advances in Artificial Intelligence, Lecture Notes in Computer Science, Springer-Verlag, pp. 998-1008, Vol. 5317.

Cetin, M. and Tesar, D., 1998, "*Performance Identification and Multi-Criteria Redundancy Resolution for Robotic Systems*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Chang, K.C., and Sun, W., 2003, "*Comparing Probabilistic Inference for Mixed Bayesian Networks*," Proc. SPIE, pp. 346-353, Vol. 5096, Signal Processing, Sensor Fusion, and Target Recognition XII, Ed. Kadar, I.

Chen, I.M, 2001, "*Rapid Response Manufacturing Through a Rapidly Reconfigurable Robotic Workcell*," Robotics & Computer Integrated Mfg., pp. 199-213, Vol. 17.

Cheng, J., and Druzdel, M.J., 2000, "*AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks,*" Journal of Artificial Intelligence Research, pp. 155-188, Vol. 13, October.

Cobb, B.R., and Shenoy, P., 2004, "*Inference in Hybrid Bayesian Networks with Deterministic Variables*," P. Lucas (ed.), Proc. of the 2nd European Workshop on Probabilistic Graphical Models, pp. 57-64, Leiden, Netherlands.

Cocca, C. and Tesar, D., 2000, "*Failure Recovery in Redundant Serial Manipulators*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Cohen, P., 1995, "*Empirical Methods for Artificial Intelligence*," MIT Press.

Cohen, I., Bronstein, A., and Cozman, F.G., 2001, "*Online Learning of Bayesian Network Parameters*," Pub. HPL-2001-55, HP Laboratories.

Colby, R.S., Mottier, F., and Miller, T.J.E., 1995,"*Vibration Modes and Acoustic Noise in a 4-Phase Switched Reluctance Motor*," Proc. 30th IAS Annual Meeting, IEEE Industry Applications Conf., pp. 441–447, Vol.1.

Connolly, M., and O'Reilly, F., 2005, "*Sensor Networks and the Food Industry*," Workshop on Real-World Wireless Sensor Networks, Stockholm, June 20–21.

Cooper, G.F., 1990, "*The Computational Complexity of Probabilistic Inference Using Bayesian Belief Networks*," Artificial Intelligence, pp. 393-405, Vol. 42.

Correa, M., Bielza, C., and Pamies-Teixeira, J., 2009, "*Comparison of Bayesian Networks and Artificial Neural Networks for Quality Detection in a Machining Process*," Expert Syst. with Applications, pp. 7270-7279, Vol. 36, No. 3, April.

Crocoll, W.M., and Coury., B.G., 1990, "*Status or Recommendation: Selecting the Type of Information for Decision Aiding*," Proc. of the Human Factors Society, 34th Annual Meeting, pp. 1524– 1528.

Dabala, K., 2001, "*Analysis of Mechanical Losses in Three-Phase Squirrel-Cage Induction Motors*," Proc. of the 5th Intl. Conf. on Electrical Machines and Systems, ICEMS-2001.

Dagum, P., and Luby, M., 1993, "*Approximating Probabilistic Inference in Belief Networks is NP-hard*," Artificial Intelligence, pp. 141–153, Vol. 60, No.1.

Darwiche, A., 2009, "*Modeling and Reasoning with Bayesian Networks*," Cambridge University Press.

Das, B., 2004, "*Generating Conditional Probabilities for Bayesian Networks: Easing the Knowledge Acquisition Problem*," in Journal CoRR cs.AI/0411034, available at arxiv.org/pdf/cs/0411034v1 (accessed in 2010).

Dean, D.A., 2006, "*Characterizing Bayesian Network Structure Sensitivity in Classifiers Derived from Data*," Masters' thesis, Department of Computer Science, The

University of Massachusetts-Lowell.

Deihimi, A., Farhangi, S., and Henneberger, G., 2002, "*A General Nonlinear Model of Switched Reluctance Motor with Mutual Coupling and Multiphase Excitation*," Electrical Engg. Journal (Springer–Verlag), pp. 143-158, Vol. 84, No.3, July.

Deml, B., Ortmaier, T., and Weiss, H., 2004, "*Minimally Invasive Surgery: Empirical Comparison of Manual and Robot Assisted Force Feedback Surgery*," Proc. of the 4th Intl. Conf. EuroHaptics, pp. 403-406, Munich, Germany.

Demling, A.G., and Tesar, D., 2006, "*Software and Test Development for Condition Based Maintenance*," Master's Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Denton, R., 2010, "*Sensor Reliability Impact on Predictive Maintenance Program Costs*," Application note, Wilcoxon Research, available at www.wilcoxon.com.

deSantana, A.L., Frances, C.R., Rocha, C.A., Carvalho, S.V., Vijaykumar, N., Rego, L.P., and Costa, J.C., 2007, "*Strategies for Improving the Modeling and Interpretability of Bayesian Networks*," Data & Knowledge Engineering , pp. 91-107, Vol. 63.

Devore, J.L, 2000, "*Probability & Statistics for Engineering & the Sciences*," Duxbury.

Diez, J.F., and Druzdel, M.J., 2007, "*Canonical Probabilistic Models for Knowledge Engineering*," Technical Report CISIAD-06-01, UNED, Madrid, Spain / Decision Systems Laboratory, University of Pittsburgh, Pittsburgh, PA, April 28.

Doell, C., 2001, "*Drive by Wire Technology*," Vehicle Technologies Symposium, Intelligent Systems for the Objective Fleet.

Doriben, H.T., and Durkopp, K., 2003, "*Mechatronics and Drive-by-Wire Systems Advanced Non-Contacting Position Sensors*," Control Engineering Practice, pp. 191–197, Vol. 11.

Drewry, M.A., and Georgiou, G.A., 2007, "*A Review of NDT Techniques for Wind Turbines*," Insight (available online at www.ndt.net/search/docs.php3?id=4550), pp. 137-141, Vol. 49, No. 3, March.

Dunia, R., and Qin, S.J., 1998, "*Joint Diagnosis of Process and Sensor Faults Using Principal Component Analysis*," Control Eng. Prac, pp. 457–469, Vol. **6**, No. 4.

Duta, M., and Henry, M., 2005, "*The Fusion of Redundant SEVA Measurements*," IEEE Trans. on Control Systems Technology, pp. 173-184, Vol. 13, No. 2, March.

Duran, F., 2003, "*Smart Equipment and Systems to Improve the Reliability and Safety in Future Nuclear Plants*," Sandia National Laboratories Nuclear Energy Research Initiative, Project No. 99-306 / PR: DE-FG03-99SF21911, August.

Ebert-Uphoff, I., 2007, "*Measuring Connection Strengths and Link Strengths in Discrete Bayesian Networks*," Tech. Report GT-IIC-07-01, Georgia Inst. of Technology.

Edrington, C.S., Kaluvagunta, D.C., Joddar, J., and Fahimi, B., 2003, "*Investigation of Electromagnetic Force Components in SR Machines: Design and Control Issues*," Proc. of the 38th IEEE Industry Applications Conf., pp. 219-226., Vol.1.

Fischer, D., Kaus, E., and Isermann, R., 2003, "*Fault Detection for an Active Vehicle Suspension*," Proc. of the American Control Conference Denver, pp. 4377-4382, Colorado, June 4-6.

Fleming, W.J., 2001, "*Overview of Automotive Sensors*," IEEE Sensors Journal, pp. 296-308, Vol. 1, No. 4, December.

Fong, T., Thorpe, C., and Glass, B., 2002, "*Robot as Partner: Vehicle Teleoperation with Collaborative Control*," in NRL Workshop on Multi-Robot Systems, Washington, DC, USA, pp. 195–202.

Fraden, J., 2004, "*Handbook of Modern Sensors: Physics, Designs and Applications*," Third Edition, AIP Press /Springer-Verlag Inc.

Frank, R., 2000, "*Understanding Smart Sensors,*" 2nd Edition, Artech House Publishers.

Freund, B., 2008, "*Real-time Monitoring of our Warfighters Health State: The Good, The Bad, and The Ugly*," Presentation at Army Telemedicine Partnerships Series 2008: Personal Health Monitoring, Seattle, WA, 5 April.

Friedl, K.E., 2007,"*Is it Possible to Monitor the Warfighter for Prediction of Performance Deterioration?*," NATO Technical Report, RTO-MP-HFM-151 , KN-7 , RTO Human Factors and Medicine Panel (HFM) Workshop held in Paris, France, 7-9 February 2007.

Friedl, K.E., and Allan, J.H., 2004, "*USARIEM: Physiological Research for the Warfighter*," Army Medical Department (AMEDD) Journal, pp. 33-43, 4Q.

Friedman, N., Linial, M., Nachman, I., and Peer, D., 2000, "*Using Bayesian Networks to Analyze Expression Data*," Journal of Computational Biology, pp. 601-620, Vol. 7, No. 3-4.

Friedman, N., Goldszmidt, M., Heckerman, D., and Russell, S., 1997, "*Challenge: What is the Impact of Bayesian Networks on Learning?*," Proc. of the 15th Intl. Joint Conf. On Artificial Intelligence, pp. 10-15, Vol. 1, Nagoya, Japan, 1997.

Frolik, J., Abdelrahman, and M., Kandasamy, P., 2001, "*A Confidence-Based Approach to Self-Validation, Fusion and Reconstruction of Quasi-Redundant Sensor Data*," IEEE Trans. on Instrumentation & Measurement, pp. 1761-1768, Vol. 50, No.6, December.

Fung, R., and Chang, K.C., 1989, "*Weighing and Integrating Evidence for Stochastic Simulation in Bayesian Networks*," Uncertainty in Artificial Intelligence 5, pp. 209-219, Elsevier Science Publishing Company, Inc, New York, N. Y.

F.W. Bell (Sypris), 2009, Catalog Information for Current Sensor NT-25, available at http://www.sypris.com/test-and-measurement/magnetics-fw-bell/nt-series.

Garrigan, N.R., Soong, W.L., Stephens, C.M., Storace, A., and Lipo, T.A., 1999, "*Radial Force Characteristics of a Switched Reluctance Machine*," Proc. of the 34th IAS Annual Meeting, IEEE Industry Applications Conf., pp. 2250-2258, Vol. 4.

Gelly, S., and Teytaud, O., 2005, "*Bayesian Networks: A Better than Frequentist Approach for Parametrization, and A More Accurate Structural Complexity Measure than the Number of Parameters*," Technical report (available at http://hal.inria.fr/docs/00/11/28/38/PDF/BN_RIA.pdf), HAL-INRIA, November.

Gieras, J.F., 1999, "*Comparison of High-Power High-Speed Machines: Cage Induction versus Switched Reluctance Motors*," IEEE Africon, pp. 675 -678, Vol.2, 1999.

Giraud, C., and Jouvencel, B., 1995, "*Sensor Selection: A Geometrical Approach*," Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS 95), '*Human Robot Interaction and Cooperative Robots*', pp. 555-560, Vol. 2.

Gobbi, R., Sahoo, N. C., and Vejian, R., 2007, "*The Measurement of Mechanical Parameters of a Switched Reluctance Motor Drive System*," Measurement Science and Technology, pp. 3636–3644, Vol. 18, No. 11.

Gopalsamy, C., Park, S., Rajamanickam, R., and Jayaraman, S., 1999, "*The Wearable Motherboard™: The first generation of Adaptive & Responsive Textile Structures (ARTS) for Medical Applications*," Virtual Reality, pp. 152-168, Vol. 4.

Griffiths, G., and Brito, M., 2008, "*Predicting Risk in Missions under Sea Ice with Autonomous Underwater Vehicles*," Proc. of IEEE/OES Autonomous Underwater Vehicles Conf., AUV-2008, pp. 1-7.

Gross, K.C., Singer, R.M., Wegerich, S.W., Herzog, J. P., Van Alstine, R., and Bockhorst, F., 1997, "*Application of a Model-Based Fault Detection System to Nuclear Plant Signals*," Proc. 9th Intl. Conf. On Intelligent Systems, Application to Power Systems, pp. 66-70, Seoul, Korea.

Gunasekaran, A. and Yusuf, Y. Y, 2002, "*Agile Manufacturing: A Taxonomy of Strategic and Technological Imperatives*," Intl. Journal of Production Research, pp. 1357-1385, Vol. 40, No. 6.

Guo, H., and Hsu, W., 2002, "*A Survey of Algorithms for Real-Time Bayesian Network Inference*," Joint AAAI-02/KDD-02/UAI-02 Workshop on Real-Time Decision Support and Diagnosis Systems.

Hajiyev, C., and Caliskan, F., 2005, "*Sensor and Control Surface/Actuator Failure Detection and Isolation Applied to F-16 Fight Dynamic*," Aircraft Engineering & Aerospace Technology: An International Journal, pp. 152–160, Vol. 77, No. 2.

Hameed, Z., Hong, Y. S., Cho, Y. M., Ahn, S. H., and Song, C. K., 2009, "*Condition*

Monitoring and Fault Detection of Wind Turbines and Related Algorithms: A review," Renewable and Sustainable Energy Reviews, pp. 1-39, Vol. 13.

Hasegawa, M., Tanah, N., Chibat,A. and Fuho,T., 2002, "*The Operation Analysis and Efficiency Improvement of Switched Reluctance Motors with High Silicon Steel*," Proc. of the IEEE Power Conversion Conf. (PCC 2002), pp. 981-986, Vol. 3.

Hatcher, E., and Tesar, D., 2000, "*Conceptual Redesign of an Advanced Prime Mover for Direct Drive Robotics and Automation Applications*," Master's Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Heckerman, D., 1995, "*A Tutorial on Learning with Bayesian Networks*," Tech. Report MSR-TR-95-06, Microsoft Corporation, March 1995 (revised November 1996).

Helson, R., 2004, "*The Compelling Case for the World's Most Used Digital Communications Protocol*," HART Communication Foundation White Paper, available at www.hartcomm.org.

Henrion, M., 1988, "*Propagating Uncertainty in Bayesian Networks by Logic Sampling*," Uncertainty in Artificial Intelligence, pp. 149-163, Vol. 2.

Henry, M.P., 1993, "*The Self-Validation Sensor: Rationale, Definitions and Examples*," Control Engineering Practice, pp. 585-610, Vol. 1, No. 4.

Heo, G., Chang, S.H., Choi, S.S., Choi, G.H, and Jae, M.H., 2005, "*Advisory System for the Diagnosis of Lost Electric Output in Nuclear Power Plants*," Expert Systems with Applications, pp. 747–756, Vol. 29.

Heredia, G., and Ollero, A., 2010, "*Virtual Sensor for Failure Detection, Identification and Recovery in the Transition Phase of a Morphing Aircraft*," Sensors, pp. 2188-2201, Vol. 10.

Honeywell Sensotec, 200, Catalog Information for Torque Sensor Model 1701-2, available online at http://content.honeywell.com/sensing/sensotec/torque.asp.

Hooper, R., and Tesar, D., 1994, "*Multi-Criteria Inverse Kinematics for General Serial Robots*," PhD. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Hunn, B.P., 2008, "*The Human Factors of Sensor Fusion*," ARL pub. no. ARL-TR-4458.

Huang, C., and Tesar, D., 2000, "*Architecture Development for the Actuator Management System Software*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Hvass, P., and Tesar, D., 2004, "*Condition Based Maintenance for Intelligent Electromechanical Actuators*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Iaconis, J., and Tesar, D., 1991, "*Parametric Design of a Robot Actuator*," Masters' Thesis, Department of Mechanical Engineering, University of Texas at Austin.

Ibarguengoytia, P., Vadera, S., and Sucar, L.E., 2006, "*A Probabilistic Model for Information Validation*," British Computer Journal, pp. 113-126, Vol. 49, No. 1.

Inamura, S., Sakai, T., and Sawa, K., 2003, "*A Temperature Rise Analysis of Switched Reluctance Motor Due to the Core and Copper Loss by FEM*," IEEE Trans. on Magnetics, pp. 1554 – 1557, Vol. 39, No. 3, May.

Jafari, R., Encarnacao, A., Zahoory, A., Dabiri, F., Noshadi, H., and Sarrafzadeh, M., 2005, "*Wireless Sensor Networks for Health Monitoring*," Proc. of 2nd Annual IEEE Intl. conf. on Mobile and Ubiquitous Systems: Networking and Services, pp. 479-481, July.

Janardhan, J., and Tesar, D., 2008, "*Test Methodology for Electromechanical Actuators*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Jensen, F.V., and Nielsen, T.D., 2007, "*Bayesian Networks and Decision Graphs*," 2nd Edition, Information Science and Statistics, Springer Verlag

Ji, J., Sun,Y., Zhu,H., and Zhao,W., 2005, "*Magnetic Field Analysis of Bearingless Switched Reluctance Motor Using Finite Element Method*," Proc. of the 8th Intl. Conf. on Electrical Machines & Systems, ICEMS 2005, pp. 2121-2123, Vol. 3.

Jiang, Li, Djurdjanovic, D., and Ni, J., 2007, "*A New Method for Sensor Degradation Detection, Isolation and Compensation in Linear Systems*," Proc. of ASME Intl. Mech. Engg. Cong. & Expo, IMECE2007, pp. 1089-1101, Vol. 9, November.

Jitnah, N., and Nicholson, A.E., 1998, "*Belief Network Algorithms: A Study of Performance based on Domain Characterization*," Lecture Notes in Computer Science, Learning and Reasoning with Complex Representations, Springer, pp. 168-187,Vol. 1359.

Jones, W.D.E, 2005, "*Easy Ride*," IEEE Spectrum Magazine, pp. 12-14, May.

Kang, S.H, and Tesar, D. 2004, "*Robust Metrology Procedures for Modular Robotic Systems Using Indoor GPS Metrology System*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Kang, C.W., and Golay, M.W., 1999, "*A Bayesian Belief Network-based Advisory System for Operational Availability Focused Diagnosis of Complex Nuclear Power Systems*," Expert Systems with Applications, pp. 21–32, Vol. 17, No.1.

Kapoor, C., and Tesar, D., 1996, "*A Reusable Operational Software Architecture for Advanced Robotics*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Kaupp, T., 2007, "*Probabilistic Human-Robot Information Fusion*," Ph.D. Dissertation, Australian Center for Field Robotics, University of Sydney.

Kenny, D.A., 2004, "*Correlation and Causality*," 2nd Ed. (http://davidakenny.net/books).

Kjaerulff, U.B., and Madsen, A.L., 2008, "*Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*," Information Science and Statistics, Ed. Jordan, M., Kelinberg, J., Scholkopf, B., Springer.

Kobayashi, F., Fukui, T., Arai, F., Fukuda, T., Kojima, F., Onoda, M., and Hotta, Y., 2002, "*Sensor Selected Fusion with Sensor Selection Based Gating Neural Network*," Proc. of the IEEE Intl. Conf. on Fuzzy Systems, pp. 1482-1487, Vol. 2.

Kokernak, J.M, and Torrey, D.A, 2000, "*Magnetic Circuit Model for the Mutually Coupled Switched-Reluctance Machine*," IEEE Trans. on Magnetics, pp. 500-507, Vol. 36, No. 2, March.

Kokkonen,T., Koivusalo, H., Laine, H., Jolma, A. and Varis, O., 2005, "*A Method for Defining Conditional Probability Tables with Link Strength Parameters for a Bayesian Network*," Proc. of Intl. Congress on Modeling and Simulation (MODSIM05), pp.428- 434, December.

Kolba, M.P., and Collins, L.M, 2006, *"Information-theoretic Sensor Management for Multimodal Sensing,"* Proc. of the IEEE Geoscience and Remote Sensing Symposium (IGARSS 2006), pp 3935-3938.

Koran, L., and Tesar, D., 2008, "*Duty Cycle Analysis to Drive Intelligent Actuator Development*," IEEE Systems Journal, pp. 453-463, Vol. 2, No. 4.

Korb, K.B., and Nicholson, A.E., 2004, "*Bayesian Artificial Intelligence*," Chapman and Hall /CRC Press.

Kreucher, K., Kastella, K., and Hero III, A.O., 2003, "*A Bayesian Method for Integrated Multitarget Tracking and Sensor Management*," ISIF, pp. 704-71.

Krishnamoorthy, G. and Tesar, D., 2005, "*A Multi-Sensor Architecture Development Framework for Intelligent Electromechanical Actuators*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Kuebler, B., Seibold, U., and Hirzinger, G., 2005, "*Development of Actuated and Sensor Integrated Forceps for Minimally Invasive Robotic Surgery*," Intl. Journal of Medical Robotics and Computer Assisted Surgery, pp. 96–107, Vol.1, No.3.

Kulkarni, A. and Tesar, D., 2009, "*Instant Center Based Kinematic and Dynamic Motion Synthesis for Planar Mobile Platforms*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Lampis, M., and Andrews, J.D., 2009, "*Bayesian Belief Networks for System Fault Diagnosis*," Quality and Reliability Engineering Intl., pp. 409-426, Vol. 25.

Lauritzen, S. and Spiegelhalter, D., 1988, "*Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems*," Journal of the Royal Statistical Society, pp. 253 – 258, Vol. B 50.

LEM Transducers Inc., 2009, Catalog Information for Voltage Sensor Model LV 25-P,

available online at www.lemusa.com.

Lewis, M., Wang, J., and Hughes, S., 2007, "*USARSim: Simulation for the Study of Human-Robot Interaction*," Journal of Cognitive Engineering and Decision Making, pp. 98-120, Vol. 1, No. 1.

Li., Z., and D'Ambrosio, B., 1994, "*Efficient Inference in Bayes Networks as a Combinatorial Optimization Problem*," Intl. Journal of Approximate Reasoning, pp. 55-81, Vol.11.

Li, C., Heinemann, P., and Sherry, R., 2007, "*Neural Network and Bayesian Network Fusion Models to Fuse Electronic Nose and Surface Acoustic Wave Sensor Data for Apple Defect Detection*," Sensors and Actuators B, pp. 301–310,Vol. 125.

Lim, S., and Cho, S.B., 2006,"*Online Learning of Bayesian Network Parameters with Incomplete Data*," Computational Intelligence, Lecture Notes in Computer Science, Springer-Verlag, pp.309-314,Vol. 4114.

Lin, H., Low, T. S., and Chen, S. X., 1996, "*Investigation on Magnetic Saturation in Switched Reluctance Motor using 2D Hybrid Finite Element Method*," IEEE Trans. on Magnetics., pp. 4317–4319, Vol. 32, No. 5, September.

Liu, Y., 2008, "*Predictive Modeling for Intelligent Maintenance in Complex Semiconductor Manufacturing Processes*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Michigan.

Liu, E., and Zhang, D., 2002, "*Diagnosis of Component Failures in the Space Shuttle Maine Engines Using Bayesian Belief Networks: A Feasibility Study*," Proc. of the 14th IEEE Intl. Conf on Tools with Artificial Intelligence (ICTAI02), pp. 181-189.

Lo, C.H., Fung, E.H.K., and Wong, Y.K., 2009, "*Intelligent Automatic Fault Detection for Actuator Failures in Aircraft*," IEEE Trans. on Industrial Informatics, pp.50-55, Vol. 5, No. 1, February.

Lunifeld, T., and Tesar, D., 1995, "*Motor Evaluation and Integration for Modular Robotic Actuators*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Magtrol Inc., 2009, Catalog Information for Hysteresis Brake Model HB 140 M-2, available at http://www.magtrol.com/brakesandclutches/hysteresis_brakes.html.

Mahalik, N., 2009, "*Processing and Packaging Automation Systems: A Review*," Sensing and Instrumentation for Food Quality and Safety, pp.12-25, Vol. 3, No.1, March.

Marques de'Sa, J.P., 2007, "*Applied Statistics*," Second Edition, Springer.

Martins, I., Esteves, J., Marques, G.D., and da Silva, F.P., 2006, "*Permanent-Magnets Linear Actuators Applicability in Automobile Active Suspensions*," IEEE Trans. on Vehicular technology, pp.86-94, Vol. 55, No. 1, January.

Masic, S., Corda, J, and Smaka, S., 2002, "*Computation of Static, Steady-state and Dynamic Characteristics of the Switched Reluctance Motor*," Automatika, pp.109−117, Vol. 43, No. 3−4.

Mast, T.A., Reed, A.T., Yurkovich, S., Ashby, M., and Adibhatla, S., 1999, "*Bayesian Belief Networks for Fault Identification in Aircraft Gas Turbine Engines*," Proc. of the 1999 Conf. on Control Applications, pp. 39-44.

Materu, P.N. and Krishnan, R., 1992, "*Estimation of Switched Reluctance Motor Losses*," IEEE Trans. on Industry Applications, pp. 668- 679, Vol. 28, No. 3.

Matsumoto, M., and T. Nishimura,T., 1998, "*Mersenne Twister: A 623-Dimensionally Equi-distributed Uniform Pseudo-Random Number Generator*," ACM Trans. on Modeling and Computer Simulation, pp. 3-30,Vol. 8, No. 1, January.

Matveev, A., 2006, "*Development of Methods, Algorithms and Software for Optimal Design of Switched Reluctance Drives*," Ph.D. Dissertation, Technische Universiteit Eindhoven.

Matveev, A., Kumichev, V., and Nilssen, T., 2003, "*Two Approaches for Modeling of Switched Reluctance Drives*," Proc. EPE-2003, Toulouse, France.

McCann, R., and Traore, W., 2008, "*Investigation of Direct Flux Measurements in Switched Reluctance Motors*," Proc. of the IEEE Power and Energy Society Gen. Mtg-Conversion and Delivery of Electrical Energy in the 21st Century, pp. 1-7.

McFarland, K.J., and Tesar, D., 2010 (expected), "*Development of Performance Maps for Warfighter Monitoring and Decision-Making*," Masters' Thesis (in progress), Department of Mechanical Engineering, The University of Texas at Austin.

Mehranbod, N., Souroush, M., and Panajapornpon, C., 2005, "*A Method of Sensor Fault Detection and Identification*," Journal of Process Control, pp. 321-339, Vol. 15.

Mehranbod, N., Souroush, M., Piovoso, M., and Ogunnaike, B., 2003, "*Probabilistic Model for Sensor Fault Detection and Identification*," AIChE Journal, pp.1787-1802, Vol. 49, No.7.

Mengshoel, O.J., Darwiche, A., and Uckun, S., 2008, "*Sensor Validation using Bayesian Networks*," Proc. of the 9th Intl. Symp. on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS-08), L.A., CA, February.

Michaelides, A., and Pollock, C., 1996, "*Reduction of Noise and Vibration in Switched Reluctance Motors: New Aspects*," Proc. of the 31st IAS Annual Meeting, IEEE Industry Applications Conf., pp. 771-780., Vol. 2.

Miller, T. J. E., 2002, "*Optimal Design of Switched Reluctance Motors*," IEEE Trans. on Industrial Electronics, pp. 15-27,Vol. 49, No. 1, February.

Miller, T.J.E., 1993, "*Switched Reluctance Motors and their Control*," Monographs in Electrical & Electronic Engg-31, Magna Physics/ Oxford Science Publications.

Miller, T.J.E., and McGilp, M., 1990, "*Nonlinear Theory of the SRM for Rapid Computer-Aided Design*," IEEE Proc. B, Electric Power Applications, pp. 337-347, Vol. 137, No. 6, November.

Mizia, D., Adamiak, K., Eastham, A. R., and Dawson, G. E, 1988, "*Finite Element Force Calculation: Comparison of Methods for Electric Machines*," IEEE Trans. on Magnetics, pp. 447-450, Vol. 24, No.1, January.

Moallem, M., Ong, C.-M., and Unnewehr, L.E., 1992, "*Effect of Rotor Profiles on the Torque of a Switched-Reluctance Motor*," IEEE Trans. on Industry Applications, pp. 364-369, Vol. 28, No. 2, March / April.

Motorsoft Inc., 2006, Specifications sheet for Switched Reluctance Motor, Model No. RA165157.

Mundt, W., Montgomery, K. N., Udoh, U. E., Barker, V. N., Thonier, G. C., Tellier, A. M., Ricks, R. D., Darling, R. B., Cagle, Y. D., Cabrol, N. A., Ruoss, S. J., Swain, J. L., Hines, J. W. and Kovacs, G. T. A., 2005, "*A Multiparameter Wearable Physiologic Monitoring System for Space and Terrestrial Applications*," IEEE Trans Info. Tech. in Biomedicine, pp. 382-391, Vol.9, No. 3, September.

Murphy, K., Weiss, Y., and Jordan, M., I., 1999, "*Loopy Belief Propagation for Approximate Inference: An Empirical Study*," UAI99, Proc. of 15th Annual Conf. on Uncertainty in Artificial Intelligence, pp. 467-475, Ed. Laskey and Prade.

Musial, W., Butterfield, S. and McNiff, B., 2007, "*Improving Wind Turbine Gearbox Reliability*," National Renewable Energy Laboratory Paper No. NREL/CP-500-41548), 2007 European Wind Energy Conference, Milan, Italy, May 7–10.

Nadkarni, S., and Shenoy, P.P., 2001, "*A Bayesian Network Approach to Making Inferences in Causal Maps*," European Journal of Operational Research, pp. 479-498, Vol. 128, No.3.

Nadkarni, S., and Shenoy, P.P., 2004, "*A Causal Mapping Approach to Constructing Bayesian Networks*," Decision Support Systems, pp. 259-281, Vol. 38, No.2.

Naish, M.D. and Croft, E.A., 1999, "*Multisensor Industrial Inspection and Grading using ELSA*," Proc. of the 1999 IEEE/ASME Intl. Conf. on Advanced Intelligent Mechatronics, pp.938-943.

Napolitano, M.R., An, Y., and Seanor, B.A., 2000, "*A Fault Tolerant Flight Control System for Sensor and Actuator Failures using Neural Networks*," Aircraft Design, pp. 103-128, Vol. 3.

National Instruments, 2009, Catalog Information for cRIO-9112, NI-9014, NI-9201, NI-9411, NI- 9472, available online at http://www.ni.com/products.

Neapolitan, R.E., "*Learning Bayesian Networks*," Prentice Hall, New York, 2003.

Neethirajan, S., Jayas, D. S., and Sadistap, S., 2009, "*Carbon Dioxide (CO_2) Sensors for the Agri-food Industry—A Review*," Food Bioprocess Tech. pp.115–121, Vol. 2.

Neil, M., Fenton, N.E., and Nielsen, L., 2000, "*Building Large-Scale Bayesian Networks*," Knowledge Engineering Review, Vol. 15, No. 3, pp.257-284.

Newman, W.S., Podgurski, A., Quinn, R.D., Merat, F.L., Branicky, M.S., Barendt, N. A., Causey, G.C., Haaser, E.L., Kim, Y., Swaminathan, J., and Velasco, V.B., 2000, "*Design Lessons for Building Agile Manufacturing Systems*," IEEE Trans. on Robotics and Automation, pp. 228-238, Vol. 16, No. 3, June.

Nicholson, A. E. and Jitnah, N., 1999, "*Using Mutual Information to Determine Relevance in Bayesian Networks*," Proc. 5th Pacific Rim Intl. Conf. on Artificial Intelligence, pp. 399-410.

Nowak, B. M., and Tesar, D., 1998, "*A Conceptual High-Resolution MR Encoder and Torque Transducer for Precision Actuators*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Odgaard, P. F., Stoustrup, J., Nielsen, R., and Damgaard, C., 2009, "*Observer Based Detection of Sensor Faults In Wind Turbines*," Proc. of European Wind Energy Conf., Mar. 16-19, Marseille, France.

O'Leary, M.D., Simone, C., Washio, T., Yoshinaka, K., and Okamura, A. M., 2003, "*Robotic Needle Insertion: Effects of Friction and Needle Geometry*," IEEE Intl. Conf. on Robotics and Automation, pp. 1774 – 1780, Vol.2, May.

Omekanda, A.M., Broche, C., and Renglet, M., 1997, "*Calculation of the Electro-magnetic Parameters of a Switched Reluctance Motor using an Improved FEM-BIEM-Application to Different Models for the Torque Calculation*," IEEE Trans. Industrial Applications, pp. 914–918, Vol. 33, No. 4, July/August.

Omekanda, A., Broche, C., Crappe, M., and Baland, R., 1998, "*Prediction of the Steady-State Performance of the Switched Reluctance Motor Using Quadratic BIEMIFEM Field Solutions in the Linear Model*," European Trans. on Electrical Power, pp. 5-12, Vol. 8., No. 1, January / February.

Omekanda, A., 2003, "*A New Technique for Multidimensional Performance Optimization of Switched Reluctance Motors for Vehicle Propulsion*," IEEE Trans. on Industry Applications, Vol. 39, No. 3, pp. 672-676, May-June.

Onel. T., Ersoy, C., and Delic, H., 2009, "*Information Content-Based Sensor Selection and Transmission Power Adjustment for Collaborative Target Tracking*," IEEE Transactions on Mobile Computing, Vol. 8, No. 8, August.

Oosterom, M., Babuska, R., and Verbruggen, H.B, 2002, "*Soft Computing Applications in Aircraft Sensor Management and Flight Control Law Reconfiguration*," IEEE Trans on Systems, Man, and Cybernetics—Part C, pp. 125-139, Vol. 32, No. 2.

O'Reilly, P.G., 1988, "*Local Sensor Fault Detection using Bayesian Decision Theory*," IEEE-UKACC Intl. Conf. on Control '98 (Publ. No. 455), pp. 247 – 251, Vol. 1.

Pal, S., and Rakshit, A., 2004, "*Development of Network Capable Smart Transducer Interface for Traditional Sensors and Actuators*," Sensors and Actuators A, No. 112, pp. 381–387.

Pallas-Areny, P, and Webster, J. G., 2001, "*Sensors and Signal Conditioning*," 2nd Ed., John Wiley and Sons Inc.

Panda, S.K., and Amaratunga, G.A.J., 1990, "*Switched Reluctance Motor Drive without Direct Rotor Position Sensing*," IEEE Industry Applications Society Annual Meeting, Vol. 1, pp. 525–530.

Pandian P.S., Safeer, K. P., Gupta, P., Shakunthala, D. T., Sundersheshu, B. S., and Padaki, V. C., 2008, "*Wireless Sensor Network for Wearable Physiological Monitoring*," Journal of Networks, pp. 21-29,Vol. 3, No. 5, May.

Parasuraman, R., Barnes, M., and Cosenzo, K., 2007, "*Adaptive Automation for Human-Robot Teaming in Future Command and Control Systems*," The International C2 Journal, Special Issue on Decision Support for Network Centric Command and Control, pp. 43-68,Vol. 1, No.2.

Parasuraman, R., Sheridan, T.B., and Wickens, C.D., 2000, "*A Model of Types and Levels of Human Interaction with Automation*," IEEE Trans. on Systems, Man, and Cybernetics – Part A: Systems and Humans, pp. 286–297, Vol. 30.

Park, S.H., and Tesar, D., 2005, "*Fundamental Development of Hypocycloidal Gear Transmissions*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Parreira, B., Rafael, S., Pires, A. J. and Branco, P. J. C., 2005, "*Obtaining the Magnetic Characteristics of an 8/6 Switched Reluctance Machine: From FEM Analysis to the Experimental Tests*," IEEE Trans. on Industrial Electronics, pp. 1635-1643, Vol. 52, No. 6, December.

Patel, P.D., 2002, "*(Bio)sensors for Measurement of Analytes Implicated in Food Safety: A Review*," Trends in Analytical Chemistry, pp. 96-115,Vol. 21, No. 2.

Pavlin, G., Maris, P. M., and Groen, F., 2007, "*Causal Bayesian Networks for Robust and Efficient Fusion of Information Obtained from Sensors and Humans*," Proc. of Instrumentation and Measurement Technology Conference, Warsaw, Poland.

PCB Piezotronics, 2009, Catalog Information for Accelerometer Model 353B17, available online at http://pcb.com/spec_sheet.asp?model=353B17.

Pearl, J., 1988, "*Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*," Series in Representation and Reasoning, Ed. Ronald J. Brachman, Morgan Kayfmann Publishers Inc.

Pillay, P., and Cai, W., 1999, "*An Investigation into Vibration in Switched Reluctance Motors*," IEEE Trans. on Industry App., pp. 589-596,Vol. 35, No. 3, May/June.

Pires, J.N., and Costa, J.M.G, 2002, "*Object-Oriented and Distributed Approach for Programming Robotic Manufacturing Cells*," IFAC Journal Robotics and CIM, pp. 29-42, Vol.16, No. 1.

Pisu, P., Soliman, A., and Rizzoni, G., 2003, "*Vehicle Chassis Monitoring System*," in Control Engineering Practice, pp.345–354, Vol. 11.

Pitzek, S. and Elmenreich, W., 2003, "*Configuration and Management of a Real-Time Smart Transducer Network*," White Paper, Institut fur Technische Informatik Vienna University of Technology, Vienna, Austria.

Pranatysato, T.N., and Qin, J.S., 2001, "*Sensor Validation and Process Fault Diagnosis for FCC units under MPC feedback*," Control Engg. Practice, pp. 877-888, Vol.9.

Pratt, W., 2004, "*Evaluating Fieldbus Networks*," White paper, www.hartcomm.org.

Podra, P., and Andersson, S., 1999, "*Simulating Sliding Wear with Finite Element Method*," Tribology International, pp. 71-81, Vol. 32.

Potter, D., 1998, "*IEEE P1451.4's Plug-and-Play Sensors*," IEEE P1451.4 Working Group and National Instruments, White Paper available at www.ni.com.

Puls, C. and Tesar, D., 1994, "*Internal Communications and Interfaces for Modular Robotics*," Master's Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Qinetiq North America, accessed 2010, TrainTrak™, http://www.qinetiq-na.com/products-commercial-traintrak.htm.

Quasdorf, J., 2003, "*BiSS: An Open Digital Interface Standard for Smart Sensors*," Application Note, iC-Haus GmbH, available at www.ichaus.com.

Qureshi, Z., and Urlings, P., 1999, "*Situation Awareness and Automation: Issues and Design Approaches*," Proc. of Information, Decision & Control (IDC 99), pp. 605-610.

Rabindran, D., and Tesar, D., 2009, "*A Differential-based Parallel Force /Velocity Actuation Concept: Theory and Experiments*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Rabindran, D., and Tesar, D., 2004, "*Preliminary Studies on Force/Motion Control of Intelligent Mechanical Systems*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Radioshack, 2009, Catalog Information for Digital Sound Level Meter Model 3-2055, available at http://www.radioshack.com/product/index.jsp?productId=2103667.

Rahman, K.M, Gopalakrishnan, S., Fahimi, B., Rajarathnam, A.V., and Ehsani, M., 2001, "*Optimized Torque Control of Switched Reluctance Motor at All Operational*

Regimes Using Neural Network," IEEE Trans. on Industry Applications, pp. 904-913, Vol. 37, No.3, May/June.

Rajabally, E., Sen, P., Steve Whittle, S., and Dalton, J., 2004, "*Aids to Bayesian Belief Network Construction*," Proc. of the 2nd IEEE Intl. Conf. on Intelligent Systems, pp. 457-461, June.

Ramos, F. T., and Cozman, F. G., 2005, "*Anytime Anyspace Probabilistic Inference*," in Intl. Journal of Approximate Reasoning, pp. 53-80, Vol. 38.

Ramu, K., 2001, "*Switched Reluctance Motor Drives- Modeling, Simulation, Analysis, Design and Applications,*" CRC Publications.

Raulin, V., Radun, A., and Husain, I., 2004, *"Modeling of Losses in Switched Reluctance Machines*," IEEE Trans. on Industry Applications, pp. 1560 – 1569, Vol. 40, No. 6, November / December.

Rensselar, J.V., 2010, "*The Elephant in the Wind Turbine*," Tribology and Lubrication Technology (Society of Tribologists and Lubrication Engineers), pp. 2-12, June.

Riascos, L.A.M., Cozman, F.G., Miyagi, P.E., and Simoes, M.G., 2006, "*Bayesian Network Supervision on Fault Tolerant Fuel Cells*," Proc. of the 41st IAS Annual Meeting, IEEE Industry Applications Conf., pp 71059 - 1066, Vol. 2.

Ricks, B. W., and Mengshoel, O. J., 2009, "*Methods for Probabilistic Fault Diagnosis: An Electrical Power System Case Study*," Proc. of the First Annual Conference of the Prognostics and Health Management Society (PHM-09), San Diego, October.

Robot Power Inc., 2009, User Manual for Open Source Motor Controller (OSMC), available online at http://www.robotpower.com/downloads/

Rolfes, R., Zerbst, S., Hakke, G., Reetz, J., and Lynch, J.P, 2007, "*Integral SHM-System for Offshore Wind Turbine Using Smart Wireless Sensors*," Proc. of the 6th Intl. Workshop on Structural Health Monitoring, Stanford, CA, September 11-13.

Ross, K., Chaney, R., Cybenko, G., Burroughs, D., and Willsky, A., 1998, "*Mobile Agents in Adaptive Hierarchical Bayesian Networks for Global Awareness*," in IEEE Intl. Conf. on Systems, Man, and Cybernetics, pp. 2207–2212, Vol. 3.

Rothenhagen, K., and Fuchs, F.W., 2008, "*Advanced Sensor Fault Detection and Control Reconfiguration of Wind Power Plants using Doubly Fed Induction Generators*," Proc. of IEEE Power Electronics Specialists Conference (PESC), pp. 913-919.

Rouhania, H., Faizb, J., and Lucas,C., 2007, "*Lumped Thermal Model for Switched Reluctance Motor Applied to Mechanical Design Optimization*," Mathematical and Computer Modeling, pp. 625–638, Vol. 45, No. 5-6, March.

Rovira, E., McGarry, K., and Parasuraman, R., 2007, "*Effects of Imperfect Automation on Decision Making in a Simulated Command and Control Task*," Human Factors, pp. 76–87, Vol. 49.

Ruspini, E.H., Chan, M., Lowrance, J., Yang, J., Murdock, J., and Yeh, E., "*Human-aided Multi-sensor Fusion*," 8th Intl. Conf. on Information Fusion, Vol. 1, 2005.

Russell, S., and Norvig, P., 1995, "*Artificial Intelligence: A Modern Approach*," Prentice-Hall.

Saric, S., Hadiashar, A.B, and Hoseinnezhad, R., 2008, "*Clamp-Force Estimation for a Brake-by-Wire System: A Sensor-Fusion Approach*," IEEE Trans. on Vehicular Technology, pp.778-786, Vol. 57, No. 2, March.

Sainsot, D.N., and Flamand, L., 1994, "Power Loss of Gear Ball Bearing under Axial and Radial Loads," Tribology Transactions, pp. 83-90, Vol. 37, No. I1.

Sass, L., McPhee, J., Schmitke, C., Fisette, P., and Grenier, D., 2004, "*A Comparison of Different Methods for Modeling Electromechanical Multibody Systems*," Multibody System Dynamics, Kluwer Academic Publishers, pp. 209-250.

Scott, E.L., and Tesar, D., 1999, "*Criteria Based Actuator Control*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin,

Shachter, R.D., and Peot, M.A, 1989, "*Simulation Approaches to General Probabilistic Inference on Belief Networks*," Uncertainty in Artificial Intelligence 5, pp. 221-231, Elsevier Science Publishing Company, Inc. New York, N. Y.

Shakkari, P., and Tesar, D., 2007, "*Math Framework for Condition-Based Maintenance of Electromechanical Actuators*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Sigwald, R., and Tesar, D., 2008, "*Geometric Analysis of the Parallel Eccentric Gear Train*," Master's Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Singh, P. C., Singh, R. K., Smith, R. S. and Nelson, P. E., 1997, "*Evaluation of In-line Sensors for Selected Properties Measurements in Continuous Food Processing*," Food Control, pp. 45-50, Vol. 8.

Soares, F., and Costa Branco, P.J., 2001, "*Simulation of a 6/4 Switched Reluctance Motor Based on Matlab/Simulink Environment*," IEEE Trans. on Aerospace and Electronic Systems, pp. 989-1009, Vol. 37, No. 3.

Spaan, T.J.M., and Lima, P.U., 2009, "*A Decision-theoretic Approach to Dynamic Sensor Selection in Camera Networks*," Proc. of 19th AAAI Intl. Conf. on Automated Planning and Scheduling, pp. 297-304.

Spiegelhalter, D. J., Franklin, R., and Bull, K., 1989, "*Assessment, Criticism, and Improvement of Imprecise Probabilities for a Medical Expert System*," Proc. of the 5th Conf. on Uncertainty in Artificial Intelligence, pp. 285-294., CA.

Speigelhalter, D.J., Dawid, A.P., Lauritzen, S.L. and Cowell, R.G., 1993, "*Bayesian Analysis in Expert Systems*," Statistical Science, pp. 219-247, Vol. 8, No.3.

Stiles, J., 2007, EECS220 course handouts, The University of Kansas, available at http://www.ittc.ku.edu/~jstiles/220/handouts.htm.

Subrahmanya, N., Shin, Y.C., and Meckl, P.H, 2010, "*A Bayesian Machine Learning-Method for Sensor Selection and Fusion with Application to On-board Fault Diagnostics*," Mechanical Systems and Signal Processing, Vol. 24, pp. 182–192.

Sudderth, E. B, Ihler, A. T., Freeman, W. T., and Willsky, A.S., 2003, "*Nonparametric Belief Propagation*," Proc. of the 2003 IEEE Conf. on Computer Vision and Pattern Recognition, pp. 605-612, Vol.1, June.

Sung, M., DeVaul, R., Jimenez, S., Gips, J., and Pentalnd, A., "*Shiver Motion and Core Body Temperature Classification for Wearable Soldier Health Monitoring Systems*," Proc. of 8th IEEE Intl. Symposium on Wearable Computers (ISWC'04), pp. 192-193, 2004.

Sungsoo Lim, and Sung-Bae Cho, 2006, "*Online Learning of Bayesian Network Parameters with Incomplete Data*," Computational Intelligence /Lecture Notes in Computer Sci., pp. 309-314, Vol. 4114, Springer Verlag.

Takahisa, K., 2003, "*Aircraft Engine Sensor/Actuator/Component Fault Diagnosis Using a Bank of Kalman Filters*," Technical Report, QSS Group, Inc., Cleveland, Ohio, NASA/CR-2003-212298, March.

Tesar, D., 2010, "*Next Wave of Technology: Technical Development in Intelligent Machines*," White paper, Robotics Research Group, The University of Texas at Austin, March.

Tesar, D., 2010 (2), "*Marriage of Man and Machine*," White paper, Robotics Research Group, The University of Texas at Austin, February

Tesar, D., 2010 (3), "*Conceptual Performance Maps for Operational Soldier Decisions*," Memo to Army Science Board (ASB) for Soldier Performance/Resilience, Summer Study Project, July.

Tesar, D., 2010 (4), "*Extended Meaning of Autonomy for Battlefield UGVs*," White paper, Robotics Research Group, The University of Texas at Austin, July.

Tesar, D., Ashok, P., Krishnamoorthy, G., Gentry, K., and Lee, H., 2010, "*Sensor Fusion and Condition Based Maintenance for Intelligent Electromechanical Actuators*," Deliverable research report, presented to the Office of Naval Research (ONR), Grant No. N00014-09-1-0427, March.

Tesar, D., 2009, "*Electro-Mechanical Actuator Architecture (EMAA)*," Internal Research Document, Robotics Research Group, The University of Texas at Austin.

Tesar, D., 2008, "*Development Timeline for proposed JLTV using Intelligent Vehicle Corner*," White paper, Robotics Research Group, The University of Texas at Austin.

Tesar, D., Ashok, P., Krishnamoorthy, G., Kendrick, K., Park. S., and Vaculik, S., 2005, "*Performance Maps for Actuator Intelligence*," White Paper, Robotics Research Group, The University of Texas at Austin.

Tesar, D., 2009, "*Electro-Mechanical Actuator Architecture (EMAA)*," White Paper, Robotics Research Group, The University of Texas at Austin.

Tesar, D., Kapoor, C., and Pryor, M., 2006, "*Open Architecture Surgical Cells based on the Trauma Pod Project*," White paper, Robotics Research Group, The Universityof Texas at Austin.

Thornhill, L., Walls, A., Arkin, R., Beno, J., Bergh, C., Bresie, D., Giovannetti, A., Gothard, B., Matthies, L., Nogueiro, P., Scanlon, J., Scott, R., Simon, M., Smith, W., and Waldron, K., 2003, "*Design of an Agile Unmanned Combat Vehicle - A Product of the DARPA UGCV Program*," Unmanned Ground Vehicle Technology V, Proc. of SPIE, pp. 358-370, Vol. 5083.

Toth, F., and Szabo, N., 2001, "*Computing the Force of Linear Machines using Finite-Element Analysis*," Proc. of Workshop on Electrical Machines' Parameters, Technical University of Cluj-Napoca, pp. 41-46, 26th May.

Turner, C.J., and Tesar, D., 2000, "*Criteria Development for Actuator Resource Management*," Master's Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

U. S. Digital Inc., 2009, Catalog Information for E6 Incremental Encoder, available online at http://usdigital.com/products/e6/.

U.S. Nuclear Regulatory Commission (USNRC), 2010, "*Nuclear Reactor Concepts Workshop Manual*," available online http://www.nrc.gov/reading-rm/basic-ref/teachers/unit3.html, accessed May 2010.

Ustun, O., 2009, "*A Nonlinear Full Model of Switched Reluctance Motor with Artificial Neural Network*," Energy Conversion and Management, pp. 2413–2421, No.50.

Vaculik, S., and Tesar, D., 2008, "*A Framework for Electromechanical Actuator Design*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Vaidya, D., Peng, J., Yang, L., and Rozenblit, J.W., 2005, "*A Framework for Sensor Management in Wireless and Heterogeneous Sensor Network*," Proc. of the 12th IEEE Intl. Conf. and Workshops on the Engineering of Computer-Based Systems (ECBS'05).

Valk, P.J.L., and Veenstra, B.J., 2009, "*Military Performance and Health Monitoring in Extreme Environments*," NATO Technical Report, RTO-MP-IST-123/MP-HFM-181-10, Proc. of Human Performance Enhancement for NATO Military Operations (Science, Technology and Ethics) Symp., October 5th-7th, Bulgaria.

Vijayraghavan, P., and Krishnan, R., 1999, "*Noise in Electric Machines: A Review*," IEEE Trans. on Industry Applications, pp. 1007-1013, Vol. 35, No. 5, September/ October.

Verron, S., Tiplica, T., and Kobi, A., 2008, "*Fault Detection with Bayesian Network*," Frontiers in Robotics, Automation & Control, pp.341-346, InTech Publ., October.

Walley, P., 1996, "*Measures of Uncertainty in Expert Systems*," Artificial Intelligence, Vol. 83, No. 1, pp. 1-58, May.

Wei, W., Dunlop, J.B., Collocott, S.J., and Kalan, B.A., 2003, "*Design Optimization of a Switched Reluctance Motor by Electromagnetic and Thermal Finite-Element Analysis*," IEEE Trans. on Magnetics, pp. 3334-3336, Vol. 39, No. 5, September.

Wei. X., Verhagen, M., and Engelen, T.V., 2009, "*Sensor Fault Detection and Isolation for Wind Turbines based on Subspace Identification and Kalman Filter Techniques*," Intl. Journal of Adaptive Control and Signal Processing, October.

Wellington, S.J., Atkinson, J.K., and Sion, R.P., 2002, "*Sensor Validation and Fusion using the Nadaraya-Watson Statistical Estimator*," Proc. 5th Intl. Conf. on Information Fusion, pp. 321–326, Vol. 1.

White, J.R., Adams, D.E., and Rumsey, M.A., 2009, "*Operational Load Estimation of a Smart Wind Turbine Rotor Blade*," Health Monitoring of Structural and Biological Systems, Ed. Kundu, T., Proc. of SPIE, pp. 72952D1-12, Vol. 7295.

Wichert, T., 2008, "*Design and Construction Modifications of Switched Reluctance Machines*," Ph.D. Thesis, Warsaw University of Technology Institute of Electrical Machines.

Winkler, C., Gordon, T., and Bareket, Z., 2005, "*Engineering Assessment of Current and Future Vehicle Technologies*," Contractor Report (DTNH22-02-D-02104), National Highway Traffic Safety Administration U.S. Department of Transportation, March.

Winquist, F., Bjorklund, R., Krantz-Rulcker, C., Lundstrom, I., Ostergren, K., and Skoglund, T., 2005, "*An Electronic Tongue in the Dairy Industry*," Sensors and Actuators- B, pp. 299–304, Vol. 111–112, June.

Wu, C.Y., and Pollock C., 1995, "*Analysis and Reduction of Vibration and Acoustic Noise in the Switched Reluctance Drive,*" IEEE Trans. on Industry Applications, pp. 91-98, Vol. 31, No. 1, January / February.

Xiong, N., and Svensson, P., 2002, "*Multi-sensor management for information fusion: Issues and approaches*," Information Fusion Vol. 3, pp. 163–186, 2002.

Xu, R. and Kwan, C., 2003, "*Robust Isolation of Sensor Failures*," Asian Journal of Control, pp. 12-23, Vol. 5, No. 1.

Yongli, Z., Limin, H., and Jinling, L., 2006, "*Bayesian Networks-based Approach for Power Systems Fault Diagnosis*," IEEE Trans. on Power Delivery, pp.634- 639, Vol. 21, No. 2, April.

Yoo, J.G. and Tesar, D., 2004, "*Actuator Performance Envelope through Nonlinear Test Bed*," Ph.D. Dissertation, Department of Mechanical Engineering, The University of Texas at Austin.

Yuh, J., 2000, "*Design and Control of Autonomous Underwater Robots: A Survey*," Autonomous Robots, pp.7–24 Vol. 8, January.

Yun, T. and Tesar, D., 2007, "*Actuator Management Operational Software (AMOS) Development for Decision Making in Intelligent Actuators*," Masters' Thesis, Department of Mechanical Engineering, The University of Texas at Austin.

Zephyr Technology, accessed 2010, BioHarness™, http://www.zephyr-technology.com/.

Zhang, R., and Bivens, A., 2007, "*Comparing the Use of Bayesian Networks and Neural Networks in Response Time Modeling for Service-Oriented Systems*," Proc. of the 2007 Workshop on Service-Oriented Computing Performance: Aspects, Issues, and Approaches, pp. 67-74.

Zhang,Y., Gu, Y., Vlatkovic, V., Wang, X., 2004, "*Progress of Smart Sensors and Smart Sensor Networks*," Proc. of the 5th World congress on Intelligent Control and Automation, June15-19, Hangzhou, China.

Zhang, S.Z., Hong, Y., Ding, H., Yang, N.H., and Wang, X.K., 2003, "*An Application of Online Learning Algorithm for Bayesian Network Parameter*," Proc. of the 2nd IEEE Intl. Conf. on Machine Learning and Cybernetics, pp. 153-156, November.

Zheng, B., Chang, Y. H., Wang, X. H., and Good, W.F., 1999, "*Comparison of Artificial Neural Network and Bayesian Belief Network in Computer-Assisted Diagnosis Scheme for Mammography*," Intl. Joint Conf. on Neural Networks (IJCNN), pp. 4181-85.

Zhou, J., and Mason, A., 2002 "*Communication Buses and Protocols for Sensor Networks*," Invited Paper, pp. 244-25, Sensors 2002, No. 2.

Zhu, W., Pekarek, S., and Fahimi, B., 2005, "*On the Effect of Stator Excitation on Radial and Tangential Flux and Force Densities in a Permanent Magnet Synchronous Machine*," Proc. of IEEE Intl. Conf. on Electric Machines and Drives, pp. 346-353.

Vita

Ganesh Krishnamoorthy received his Bachelor of Engineering (B.E.) in Mechanical Engineering from The Veermata Jijabai Technological Institute (formerly, Victoria Jubilee Technical Institute), University of Mumbai, India, in 2002. After joining the doctoral program at the Robotics Research Group in the Department of Mechanical Engineering (ME), he received his Master of Science (M.S.E.) degree in ME from The University of Texas at Austin in 2005. During his stint at Intel Research Seattle in 2006, he was a member of a team that successfully demonstrated the first use of electric field sensing for human-robot interaction. He was awarded the Continuing Fellowship instituted by the University of Texas Graduate School in recognition of research and academic excellence, in 2007-08 and was also the ME department nominee for the IBM Ph.D. fellowship in 2008-09. His research interests include sensing and decision making in intelligent mechanical systems and design of complex electromechanical systems.

Permanent Address: 3543, Greystone Dr., Austin, TX 78731.

This dissertation was typed by the author.